VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2020 Matej Tomáš

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Matej Tomáš**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2601R013 Telekomunikační technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování:

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: XEVOS SOLUTIONS, s.r.o.
2. Struktura závěrečné zprávy:
a. Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta
b. Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti
c. Zvolený postup řešení zadaných úkolů
d. Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe
e. Znalosti či dovednosti scházející studentovi v průběhu odborné praxe
f. Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení

Seznam doporučené odborné literatury:

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Zdeňka Chmelíková, Ph.D.**

Konzultant bakalářské práce: Adam Koudela

Datum zadání: 01.09.2019
Datum odevzdání: 30.04.2020

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky 1D16 OSTRAVA

prof. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 14. května 2020 .....

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu prostudium v bakalářských programech VŠB-TU Ostrava

V Ostravě 15. května 2020 .....

Týmto by som sa chcel poďakovať firme XEVOS SOLUTIONS, s. r. o. za všetku pomoc, čas a hlavne za príjemné a ústretové prostredie, mne poskytnuté pri tvorbe tejto práce. Ďalej by som sa chcel poďakovať vedúcej práce Ing. Zdeňke Chmelíkovej Ph.D. za poskytnuté konzultácie a rady pri spracovávaní tejto práce.

Abstrakt

Táto práca je spracovaním môjho absolvovania individuálnej odbornej praxe vo firme XEVOS SOLUTIONS, s.r.o. so zameraním na kybernetickú bezpečnosť. Zameraním tejto práce je opísať metodiku, prevenciu a rôzne spôsoby testovania webovej aplikácie vyvíjanej pre komerčné použitie. Všetky použité nástroje sú založené na báze open source. Bezpečnostné riziká a metodiky sú zostavené podľa vzoru nadácie OWASP. K jednotlivým rizikám sú doplnené ukážky možných útokov ako aj prevencia proti nim. V teoretickej časti práce budú popísané niektoré základné, ale kľúčové pojmy k upresneniu problematiky a písaniu tejto práce. Praktická časť bude zameraná na testovanie webového rozhrania primárne pomocou aplikácie OWASP ZAP ale aj iných open source nástrojov k rôznym účelom ako napríklad zhromažďovanie informácií a zraniteľností k vypracovaniu útočných vektorov.

Kľúčové slová: kybernetická bezpečnosť, bezpečnostné riziká, OWASP, penetračné testovanie, prevencia, open source

Abstract

This bachelor thesis is the processing of completion of my individual professional practice in XEVOS SOLUTIONS, s.r.o. with a focus on cyber security. The aim of this work is to describe the methodology, prevention and various ways of testing a web application developed for commercial use. Used tools are based on open source. Security risks and methodologies are modeled on the basis of OWASP Foundation. Individual risks are accompanied by demonstrations of possible attacks as well as prevention against them. The theoretical part of the thesis will describe some of the basics, but key concepts to clarify the issue and write this thesis. The practical part will be focused on testing the web interface primarily using the OWASP ZAP application but also other open source tools for various purposes such as gathering information and vulnerabilities to develop attack vectors.

Keywords: cyber security, security risks, OWASP, penetration testing, prevention, open source

Obsah

Zoznam použitých skratiek a symbolov

API	–	Application Programming Interface
CORS	–	Cross Origin Resource Sharing
CSRF	–	Cross-site request forgery
DNS	–	Domain Name System
DOM	–	Document Object Model
GDPR	–	General Data Protection Regulation
GUI	–	Graphical User Interface
HTML	–	Hyper Text Markup Language
HTTP	–	Hypertext Transfer Protocol
IPv6	–	Internet Protocol Verzia 6
IP	–	Internet Protocol
ISSAF	–	Information Systems Security Assessment Framework
IT	–	Information Iechnology
MAC	–	Media Access Control
NAT	–	Network address translation
OSSTMM	–	Open-Source Security Testing Methodology Manual
OS	–	Operačný Systém
OWASP	–	Open Web Application Security Project
PC	–	Personal Computer
RMB	–	right mouse button
SQL	–	Structured Query Language
SSL	–	Secure Sockets Layer
TCP	–	Transmission Control Protocol
TLD	–	Top Level Domain
TTL	–	Time to Live
UDP	–	User Datagram Protocol
URL	–	Uniform Resource Locator
VPN	–	Virtual Private Network
WSL	–	Windows Subsystem for Linux
XSS	–	Cross-site Scripting
ZAP	–	Zed Attack Proxy
csv	–	Comma-separated Values
json	–	JavaScript Object Notation
v3	–	Verzia 3.x
xlsx	–	Microsoft Excel Open XML Spreadsheet
xml	–	eXtensible Markup Language

Zoznam obrázkov

Zoznam tabuliek

Úvod

Penetračné testovanie alebo etnický hacking. Toto je svet o ktorom som vždy vedel ale zároveň nemal žiadnu predstavu. Nikdy som nečakal, že raz sa stanem jeho súčasťou a možno sa, on neskôr, stane mojou budúcnosťou.

Moje štúdium na vysokej škole mi dalo možnosť zoznámiť sa s rôznymi odvetviami informačných technológií. Celý tento čas som ale túžil o možnosť preklopenia týchto vedomostí a znalostí do reálneho sveta práce. Táto možnosť dostať do sveta informatiky, už počas štúdia, sa mi naskytla po mojej iniciatíve v projekte - Absolvovanie individuálnej odbornej praxe vo firme XEVOS Solutions, s.r.o. na pozícií Cyber security geek teda v oblasti kyber bezpečnosti.

Zabezpečenie a teda bezpečnosť ako taká, bola a je pre človeka podstatnou súčasťou života. V dnešnom svete digitalizácie a automatizácie sa zvyšuje riziko straty integrity, či už u jednotlivca alebo spoločnosti, prelomením bezpečnosti. Tak ako v reálnom tak aj v digitálnom svete sa nachádzajú medzery a slabiny, ktoré vznikli z nedostatku či už financií, pozornosti alebo znalostí. Tieto slabiny sa dajú zneužiť. Toto digitálne ohrozenie spôsobujú hackery za účelom vlastného obohatenia alebo napáchať škodu. Avšak táto činnosť sa dá vykonávať aj s iným zámerom. Zámerom prevencie a zvýšenia bezpečnosti systémov a údajov klientov. Ľudí zaoberajúcich touto činnosťou nazývame etnický hackery.

Pri mojej praxi vo firme som mal možnosť do tohto sveta nahliadnuť. Dostal som možnosť sa vzdelávať a zúčastniť na práci na projektoch pre reálne firmy po celej Českej Republike. Samozrejme možnosti tohto odvetvia sú početné a hĺbky internetu v nedohľadne. Avšak aj za tú krátku chvíľu som pochopil, že napriek rýchlemu a nezastavitelnému pokroku technológií, bezpečnosť často ostáva prehliadnutá, najmä v súkromných firmám mimo IT odvetvia. A že niektoré systémy sa dokážu prelomiť aj pri použití jednoduchých nástrojov, ktoré sú dostupné a jednoducho použiteľné pre každého kto tomu venuje trochu svojho času a energie.

Táto práca je spracovaním niektorých mojich, v tejto krátkej dobe nadobudnutých, schopností a skúsenosti v tomto odvetví. Kvôli obsiahlosti tejto problematiky sa tu nachádzajú len niektoré prístupy a metódy testovania systémov a taktiež moje výstupy pri práci na nich, a pri mojej úprimnej snahe tieto slabiny systému öpraviťä zabrániť v prelomení bezpečnosti. Avšak musíme pochopiť, že žiadna obrana nie je dokonalá. Avšak, v niektorých prípadoch, dokážeme zvýšiť obtiažnosť hackingu tak, aby sme odradili aj skúseného útočník.

1 Odborná prax

1.1 O spoločnosti

Spoločnosť XEVOS Solutions, s.r.o. (pobočka Ostrava) poskytuje komplexné IT riešenia - od systémovej integrácie, servisu a podpory, cez cloudové, serverové, sieťové a tlačové riešenia, až po dodávky hardwarového a softwarového vybavenia, pri využití platforiem PC alebo MacOS.

Medzi primárne aktivity spoločnosti patrí predovšetkým IT podpora a servis hardwarových a softwarových riešení. Tieto služby sa poskytujú jednak na vlastných pobočkách ale hlavne ön-site"teda priamo u zákazníka. V oblasti servisu PC zariadení, periférií a komponentov vytvorili servisné centrum, v ktorom sa zabezpečujú opravy všetkých bežne dostupných značiek na trhu, v záručnej či pozáručnej lehote.

Cieľom spoločnosti je pomáhať organizáciám i jednotlivcom zvyšovať efektivitu práce a chrániť ich činnosť, a podnikanie. V posledných rokoch sa začali špecializovať v oblasti kybernetickej bezpečnosti a s tým súvisiacou ochranou zákazníkov.

Všetky implementácie a odborné práce realizované spoločnosťou XEVOS, vykonávajú odborne vyškolení a certifikovaný špecialisti.

1.2 Pracovné zaradenie

Oddelenia ku ktorým som bol počas mojej praxe zaradený boli Development a HelpDesk.

Development sa zaoberá návrhom, vývojom a implementáciou rôznych softwarových riešení. Príkladom môže byť návrh, vývoj a nasadenie webových aplikácií, serverových riešení, automatizácia alebo virtualizácia.

HelpDesk je oddelenie pracujúce ako zákaznícka podpora a servis serverových, cloudových, sieťových, bezpečnostných a hardwarových riešení. Tieto činnosti sa vykonávajú na pobočkách, online alebo, najčastejšie on-site, u klienta.

Moja pracovná náplň bola zahrnutá v oboch týchto oddeleniach. Dôvodom tohto zaradenia bola jednak moja vlastná iniciatíva so zoznámením sa s rôznymi odvetviami IT v praxi ako aj nutnosť dovzdelania sa v aplikovaných odvetviach pri výkone v praxi. Podieľal som sa na niekoľkých činnostiach vo firme.

1.3 Náplň a popis práce

1. Scriptovanie príkazov pre implementáciu, správu a monitorovanie činnosti virtuálnych strojov v programe Microsoft Hyper-V, ktorých vzdialená správa a implementácia bola založená na technológií "Clouder", vyvíjanou spoločnosťou XEVOS. Následná implementácia skriptov do statického kódu programu v rozhraní WEB API, s použitím jazyka .NET Core v3 pre prostredie Visual Studio Code.

Vďaka predchádzajúcim skúsenostiam v prostredí Hyper-V, bola práca v rozsahu mojich schopností. Náročnejším prvkom práce bolo optimalizovanie vytvorených scriptov a zoznámenie sa s novým editačným prostredím Visual Studio Code. Časová náročnosť - mierna.

2. Práca s interným systémom firmy pre integráciu dát z databázy portálu HEUREKA do firemnej databázy web storu. Mojou úlohou v tejto časti bolo zoznámiť sa, pracovať s týmto systémom a vytvárať kategórie podľa požiadavkov vedenia. Následná špecifikácia kategórií a úprava ich vzoru k čo najväčšej podobnosti portálu HEUREKA za účelom čo najlepšej integrácie dat. Následné vytváranie regulárnych výrazov z dôvodu filtrácie dat pre správnu kategorizáciu a integráciu týchto databáz.

   Práca s interným systémom bola, vďaka jeho prehľadnosti, veľmi dobre zvládnuteľná. Miernu prekážku nastavilo vytváranie regulárnych výrazov(regex) avšak po kratšom zoznámení neboli problémom. Časová náročnosť - nízka.

3. Spoluúčasť na kompletnom vývoji webovej aplikácie. Práca v týme programátorov pod vedením team leadera. Zaučenie sa a práca s workflow aplikáciami ako Youtrack a Git. Vývoj šablón pre webové prostredie, tlačítka a ich funkčná logika, tvorba formulárov, správne prepisovanie údajov z a do databázy, implementácia logovacieho systému, konvertovanie tlačív do rôznych formátov (pdf, xlxs, docx, ...), error handling, a iné. Na vývoj (backend) bol použitý jazyk C-sharp, .NET Core v3 a prvky Javascriptu.

   Účasť na vývoji aplikácie bola výborná skúsenosť. Avšak kvôli rozdielom v obore a množstvu nových (pre moju osobu) programov, programovacích jazykov, postupov a veľkosti projektu to bola náročnejšia časť tejto praxe. Časová náročnosť - veľmi vysoká.

4. Práca na testovaní a penetračnom testovaní systémov a aplikácií. Práca prebiehala po zaškolení a pod odborným dohľadom. Špecifikácia testov podľa požiadavkov klienta. Prevedenie testov a spracovanie výsledkov. Na testovanie sa používali výhradne open source nástroje ako OWASP ZAP a špecializované systémy Linuxu (napríklad Kali Linux). Možnosť použitia licencovaných nástrojov, ako Burp Suite, avšak kvôli potrebe úplného zverejnenia práce budú uvedené len open source nástroje.

   Časovo náročná časť praxe z dôvodu práce s novými nástrojmi, postupmi a problematikami. Náročnosť sa zvyšovala aj kvôli rôznym prístupom práce z ktorých niektoré požadovali vyššiu investíciu času. Časová náročnosť - veľmi vysoká.

2 Teoretický úvod práce

K tomu aby som mohol vypracovať zadané činnosti, bolo nutné zvládnuť teóriu týchto odvetví. V tejto práci sa budem bližšie zaoberať, mnou vybranou, problematikou penetračného testovania, ktorá mi bola najbližšia. Táto kapitola slúži k stručnému priblíženiu niektorých pojmov a teórie o tomto odvetví.

2.1 Úvod do penetračného testovania

Penetračné testovanie (nazývané aj pentest alebo etické hackovanie) je oprávnený a systematický proces odhaľovania zraniteľných miest a hrozieb v sieťach, systémoch a aplikáciách. Ide o kontrolovanú formu hackerstva, pri ktorej sa tester snaží simulovať postup „útočníka“, aby odhalil a otestoval zraniteľnosti, ktoré by mohli byť zneužité. [1] Tieto zraniteľnosti sú dôsledkom viacerých faktorov ako napríklad zlá alebo nesprávna konfigurácia, známe a neznáme hardwarové/softwarové chyby alebo nedostatky v technický opatreniach. Odborníci v oblasti bezpečnosti napodobnia techniky používané útočníkmi, ale bez toho, aby spôsobili škodu. Výstupná správa (teda report) týchto testov slúži ako informácia k implementácií potrebných bezpečnostných opatrení. Vykonanie bezpečnostného testu v počítačových systémoch je nevyhnutné pre bezpečnosť každej organizácie vzhľadom na to, že narušenie alebo kompromitovanie prístupu do systému a k dátam môže spôsobiť finančnú ujmu ako aj ujmu na mene organizácie.

2.1.1 Čo môžme testovať?

Je možné testovať rôzne druhy technológií. Medzi najbežnejšie druhy testovania sa môžu zaradiť:

- Testovanie sietí
- Testovanie cloudových technológií
- Testovanie mobilných a webových aplikácií
- Sociálne inžinierstvo

Aj keď každá oblasť penetračného testovania má odlišné sady nástrojov, väčšinou zdieľajú spoločnú metodológiu.

2.2 Metodológie penetračného testovania

Penetračné testovanie pozostáva z rôznych, systematicky zostavených, fáz v závislosti na type vykonávaného testovania a zvolenej metodológie. Za metodológiu, môžme v tomto prípade považovať, sadu pravidiel a postupov podľa ktorých by sme mali postupovať. Ide teda o systematickú osnovu, ktorá dodáva testovaniu štruktúru. Metodológií, alebo teda štandardov, existuje

mnoho druhov. Najznámejšími sú ISSAF, OSSTMM a OWASP. Keďže pôjde o testovanie webovej aplikácie najbližším výberom je metodológia OWASP. Všeobecne by sme však postup mohli charakterizovať v 6 krokoch:

1. Plánovanie a príprava
2. Získavanie informácií
3. Posudzovanie zraniteľnosti
4. Pokus o zneužitie systému
5. Analýza zraniteľností
6. Dokumentácia (report)

Metodológie nám pomáhajú rozložiť úlohu na menšie celky o určitých spoločných vlastnostiach, a zároveň dávajú vykonávaným postupom určitý logický smer. Teraz si priblížime časti samotné.

2.3 Plánovanie a príprava

Vo fáze plánovania je dôležitým krokom kontakt a dohoda s klientom. Komunikácia je pri testovaní bezpečnosti kľúčovým bodom vzhľadom na rôzne úrovne citlivosti dát, prípadné nedokonalosti v systéme alebo pri snahe predísť nedorozumeniam a prípadným sporom so zákonom. Pri komunikácií s klientom zistíme, aký typ testu musíme zvoliť aby sme naplnili jeho účel. Typov a dôvodov prečo ich delíme, a používame je niekoľko.

2.3.1 Typy testovania podľa prístupu

„Existuje niekoľko spôsobov, ako vykonať penetračný test. Klient môže požadovať, aby tester vykonal externý penetračný test, ktorý simuluje útok z internetu. Pentester vykonáva útoky na strane klienta a sociálne inžinierstvo, aby získal prístup k internej sieti klienta. Niektoré testy vyžadujú, aby pentester vykonal interný penetračný test, kde sa chová ako nebezpečný zamestnanec alebo útočník, ktorý už obvod porušil. Klient môže taktiež požadovať, aby pentester otestoval bezpečnosť bezdrôtovej siete.“ [2]

2.3.1.1 Black box prístup, alebo čierna skrinka, je metóda testovania softvéru, ktorá analyzuje funkčnosť softvéru/aplikácie bez toho, aby tester mal akékoľvek znalosti o vnútornej štruktúre / dizajne testovaného prostredia. Tester je zodpovedný za zhromažďovanie informácií o cieľovej sieti alebo systéme. Ide o realistický scenár útoku pri ktorom "útočník"musí detailne odhaľovať a mapovať infraštruktúru a chovanie systému v rôznych stavoch. Tento typ prístupu môže byť, podľa množstva pridelených zdrojov, časovo najnáročnejší alebo zároveň najmenej náročný. Nevýhodou black boxu je, že niektoré oblasti, prvky a aplikácie môžu zostať neotestované pretože neboli odhalené.

2.3.1.2 White box je spôsob testovania vnútornej štruktúry, návrhu a statického kódu softvérového riešenia. Pri tomto type testovania má tester priamy prístup k statickému kódu. Zameriava sa predovšetkým na overenie vstupov a výstupov prostredníctvom aplikácie, zlepšenie dizajnu, použiteľnosti a posilnenie bezpečnosti. Tento spôsob testovania je výhodný v tom, že dokáže vytvoriť pevné a bezpečné vnútorné prostredie. Avšak kvôli množstvu sprístupnených dat môže byť časovo veľmi náročný.

2.3.1.3 Grey box je technika na testovanie softvérového produktu alebo aplikácie s čiastočnými znalosťami vnútorných funkcií aplikácie alebo základnými prístupovými právami. Účelom tohto testovania je hľadať chyby spôsobené nesprávnou štruktúrou kódu alebo nesprávnym fungovaním / používaním aplikácie.V tomto procese sa zvyčajne identifikujú chyby súvisiace s kontextom, ktoré súvisia s webovými systémami. Gray box test je metóda testovania softvéru, ktorá je kombináciou testovania white a black box.

Po zvolení správneho typu testovania, ostáva vybrať typ prevedenia testovania. Tento krok väčšinou volí sám tester podľa svojich preferovaných schopností a uváženia, avšak nie je netradičným, aby mal klient svoje vlastné, a konkrétne požiadavky. Každý typ prevedenia testu má svoje plusy a mínusy.

2.3.2 Typy testovania podľa prevedenia

2.3.2.1 Automatizované testy majú výhodu vysokej rýchlosti, efektivity a spoľahlivosti. Tieto nástroje sú vyvíjané profesionálmi tak, že nevyžadujú vysoké nároky na použitie. Dokážu veľmi rýchlo poskytnúť veľké množstvo dát. Ide o overené, zautomatizované postupy manuálneho testovania. Ich slabou stránkou je nemožnosť odhaliť komplikované zraniteľnosti ako napríklad chyby obchodnej logiky. Odhaľujú hlavne bežné a najpočetnejšie zraniteľnosti.

2.3.2.2 Manuálne testy závisia výlučne od schopností testera. Tento prístup je najbežnejšou praxou, pretože prináša viac odhalených zraniteľností v obchodnej logike než bežných zraniteľností, ktoré môžu nájsť automatizované nástroje. Tento prístup je časovo náročný a nákladný. Pre klienta je však najvýhodnejší pri hľadaní slabín obchodnej logiky, ktorému žiadny automatizovaný nástroj nedokáže oponovať. Nevýhodou je, že neskúsený tester môže prehliadnuť zraniteľné miesto a to môže byť dodatočne zneužité.

2.3.2.3 Semiautomatizované testy sú *„kombináciou automatických a manuálnych testov. Pri tomto type testov sa najskôr odhaľujú možné zraniteľnosti pomocou testov automatických a na základe toho sú,pre konkrétne zraniteľnosti, zvolené testy manuálne. “* [3]

2.4 Zber informácií

Ďalším nasledujúcim krokom metodológie je zber informácií. Zber informácií je pre útočníka kritický. Pokiaľ chce byť úspešný vo svojej snahe, je vyžadovaná dokonalá, alebo čo najlepšia,

znalosť cieľa. *„Hlavným cieľom tejto fázy je analyzovať sieťové rozmiestnenie IP adries, názvy serverov, aplikácie, ktoré udržiavajú databázu, kontaktné informácie a všetky možné zraniteľnosti založené na softvéri. “* [4] Toto množstvo zhromaždených informácií nám umožňuje rozšíriť možné útočné vektory a tým zvyšuje pravdepodobnosť úspechu. Najviac bežnými a užitočný rutinnými informáciami sú :

- Emailové adresy
- Telefónne čísla
- Systémové informácie
- Pracovné miesta
- Životopisy
- Osobné údaje

Najlepším a asi najrozšírenejším spôsobom kolekcie týchto informácií je scanovanie.

2.4.1 Scanovanie

Cieľom scanovania je získať čo najväčšie množstvo informácií o cieli prostredníctvom interakcie s ním, a to je jedna z najdôležitejších fáz penetračného testovania, pretože stanovuje a definuj základy, na ktoré sa dá zamerať ďalej. Počas skenovania sa chceme dozvedieť viac o topológii cieľového prostredia, pretože nám to pomôže pochopiť, ako sú rôzne systémy v prostredí navzájom prepojené a to nám dá základný obraz o plánovaní a možnostiach útoku. Taktiež sa môžme dozvedieť o operačnom systéme a parametroch serveru na ktorom dané zariadenia/aplikácie pracujú. Scany delíme na dve kategórie.

2.4.1.1 Pasívny scan napomáha pri snahe pochopiť logiku testovanej aplikácie. Ide o zber informácií pomocou rôznych nástrojov, ktorých činnosť sa priamo nedotkne cieľa. Ide o analýzu HTTP záhlavia požiadavkou a odpovedí, online výskum pozadia klienta a prostredia s využitím internetového priestoru tj. články, webové stránky, sociálne média a iné zdroje. Predchádzanie penetračnému testu pasívnym skenovaním jednoznačne zvýši šance úspešnosti testov.

2.4.1.2 Aktívny scan slúži k aktívnemu zhromažďovaniu informácií, ktorým získame informácie o danom cieli, prostredníctvom aktívnej interakcie s ním. Môže ísť o DNS Enumeration, Port Scanning, OS Fingerprinting. Podobne ako pri pasívnom prístupe, cieľom je získať čo najväčšie množstvo informácií. S týmto prístupom musí byť klient (cieľ) oboznámený pretože v opačnom prípade môže ísť o nelegálnu činnosť.

Po dokončení fázy zberu informácií, nasleduje ďalší.

2.5 Posudzovanie zraniteľností

V tejto fáze začne tester aktívne objavovať a dokumentovať rôzne zraniteľné miesta. Každému nájdenému riziku je priradená priorita, v závislosti na pravdepodobnosti jeho úspešného zneužitia. Zraniteľnosti sa odhaľujú a posudzujú pomocou analýzy predošle získaných informácií, použitím automatizovaného, aktívneho scanu zraniteľnosti a manuálnou kontrolou. Manuálna analýza a kontrola je veľmi dôležitou a efektívnou súčasťou analýzy, avšak jej výsledky závisia, v priamej úmere, na schopnostiach a kritickom myslení testera. Následne je vhodné daným zraniteľnostiam priradiť kategóriu závažnosti. Metodológia OWASP, používaná v tejto práci, kategorizuje riziká do 3 kategórií a to: nízka, stredná, vysoká. Táto kategorizácia prebieha na základe vyhodnotenia rozličných elementov ako potrebná úroveň schopností k zneužitiu, obtiažnosť odhalenia zraniteľnosti, možný dopad na systém po zneužití a mnoho iných. Väčšina nástrojov, ktoré používajú automatizovaný scan zraniteľností, si tieto úrovne vyhodnocujú samé.

Po analýze zraniteľností a rizík sa tester posunie ku praktickému výkonu testu.

2.6 Zneužitie zraniteľností systému

Zneužitie zraniteľností systému je fáza kde sa tester snaží zneužiť nájdené zraniteľnosti a slabiny, z predchádzajúcich krokov, proti ochrane systému alebo aplikácie. Cieľom tejto fázy je získanie prístupu alebo právomocí k systému a dátam. Táto fáza, by sa dala považovať, za najnáročnejšiu z pohľadu nárokov na schopnosti testera. K zneužitiu tester využíva rôznych nástrojov, existujúce postupy alebo aj vlastné kódy a skúsenosti. Zneužitia zraniteľností je možné testovať samostatne, avšak pri penetračnom testovaní je nutné otestovať čo najväčšie množstvo systémov, vstupov a útočných vektorov. Preto je vhodné použitie špecializovaných nástrojov, ktoré nám sledujú a ukladajú postup a získané dáta počas testu, ako napríklad OWASP ZAP, Metasploit alebo Burp Suite.

Po dokončení testovania, sa tester zameria na analýzu a výklad nájdených zraniteľností, a to aj takých ktoré zneužité neboli.

2.7 Analýza zraniteľností

To, že sa testerovy nepodarilo zneužiť všetky zraniteľnosti neznamená, že sa to nepodarí niekomu inému. V tejto fáze sa analyzujú nájdene bezpečnostné riziká, špecifikuje sa ich obsah a dodajú sa odporúčania na ich odstránenie.

Posledným bodom a zároveň veľmi dôležitým je podať hlásenie o postupoch a nálezoch, a to je zdokumentovanie práce.

2.8 Dokumentácia

Poslednou a záverečnou fázou, penetračného testovania, je konštrukcia výslednej správy, reportu. Report je správa, ktorá poskytuje pohľad na priebeh vykonávaného testu a jeho výsledkoch. Táto

výsledná správa je v podstate to, za čo klient platí. Pri písaní správy by sa mal tester ubezpečiť, že technické zistenia, nálezy a odporúčania sú organizované a zmysluplné, a správa ako taká je prehľadná, jasná a pochopiteľná pre klienta.

Po objasnení kľúčových pojmov a metodológie tejto práce je vhodné prejsť na praktickú ukážku.

3 Praktické riešenie úlohy penetračného testovania

V tejto časti práce sa budem venovať mojej práci a postupom na projekte pre spoločnosť XEVOS Solutions, s.r.o. Tento projekt bol vyvinutý a funkčne nasadený do prevádzky, na zákazku pre klienta. Z dôvodu GDPR, požiadaviek firmy a pri snahe dodržať pravidlá úplného zverejnenia práce boli mená, objekty a prípadné identifikačné prvky potrebne pozmenené. Zmena identifikačných prvkov nijak neovplyvní funkčnosť práce.

Táto práca sa riadi metodológiou OWASP opísanou v sekcií v 2.2, a podľa nej je štruktúrovaná aj praktická časť tejto práce. Na konci tejto časti bude uvedená konfigurácia použitých programov k správnej funkčnosti. A teraz prejdime k prvej, praktickej, časti.

3.1 Plánovanie a príprava

V prvej fáze prípravy si musíme určiť niekoľko kľúčových bodov. V tejto fáze bolo s klientom dohodnuté, že systém sa bude testovať ešte pred reálnym nasadením na östrý"server. Testovaním prostredím je teda webová aplikácia. Cieľom je otestovanie bezpečnostných prvkov, prípadné zneužitie a zistenie rozsahu možného zásahu a eskalácie útoku. Test je preventívnym otestovaním bezpečnosti, ktorý bude slúžiť k oprave identifikovaných bezpečnostných slabín.

K testu bol poskytnutý 1 prihlasovací účet a dočasná testovacia doména. Poskytnuté údaje klasifikujú typ prístupu ako gray box, vzhľadom k vlastníctvu účtu. Vykonávané testy budú ako automatizované tak aj manuálne v závislosti od potreby. Aj napriek poskytnutému účtu, bude prebiehať kontrola bezpečnosti prihlasovania a obtiažnosti získania týchto údajov. Ďalším cieľom testovania bude vnútorné prostredie aplikácie a možnosti zneužitia, vytvorenia škody.

K testovaniu aplikácie bude použitý program OWASP ZAP. K získaniu prihlasovacích informácií, údajov o prevádzkovanom servery a domén, boli vybrané tieto nástroje:

- Recon-ng
- Dmitry
- (Ze)Nmap
- TheHarvester

Po ujasnení cieľu, poskytnutých informácií a nástrojov je na rade ďalší krok, ktorý nám rozšíri, nami tvorenú a požadovanú, informačnú databázu.

3.2 Zber informácií

Zber informácií je pre útočníka kritický. Pokiaľ chce byť úspešný vo svojej snahe, je vyžadovaná dokonalá, alebo čo najlepšia, znalosť cieľa. *„Hlavným cieľom tejto fázy je analyzovať sieťové rozmiestnenie IP adries, názvy serverov, aplikácie, ktoré udržiavajú databázu, kontaktné informácie a všetky možné zraniteľnosti založené na softvéri. “* [4]

Nižšie je uvedený krátky úvod do nástrojov, ktoré boli použité pri procese výkonu tejto práce. Z dôvodu rozsiahlosti nástrojov nebudú popísané všetky funkcionality a možnosti.

3.2.1 Nástroj Recon-ng

Recon-ng je plnohodnotný, open source, nástroj pre webový prieskum predstavený ako súčasť systému Kali Linux 2. Používa rôzne moduly a interakciu s databázou čo poskytuje veľmi výkonné a jednoduché pracovné prostredie. [9]K jeho použitiu musíme mať k dispozícií systém Kali Linux alebo ak používame Windows musíme zabezpečiť inštaláciu Windows Subsystem for Linux, ktorá je úplne zdarma, a nainštalovať Kali Linux.

3.2.1.1 Prácu v recon-ng započneme pomocou terminálu, prepnutím do adresára s inštaláciou a zadaním príkazu *„sudo recon-ng“*, a zadaní hesla. Root práva sú nevyhnutné, bez nich by nám program neposkytoval jeho plnú funkcionalitu. Pri použití WSL spustíme príkazový riadok s administrátorskými právami a zadáme príkaz *„kali“*. Od tohto momentu pokračujeme ako v systéme Linux. Dáta sa ukladajú do databázy a kvôli prehľadnosti dát rôznych projektov, ako prvý vytvoríme workspace, s názvom Example. Použijeme príkaz *„workspaces create Example“*, čo nás po vykonaní prepne do vytvoreného workspacu. Ďalším krokom je nainštalovať moduly, ktoré budeme používať. Mali by sme si dopredu naplánovať čo chceme vyhľadať a s čím budeme pracovať. Avšak pre jednoduchosť môžme nainštalovať všetky moduly naraz - *„marketplace install all“*. Po inštalácií máme niekoľko možností. Mnou zvolená metodika má diagram *„domény -> profily -> kontakty “*. Pre vyhľadanie hostov domén použijeme moduly `bing_domain_web`, hackertarget a certificate transparency. Pre profily a kontakty profiler, Hunter IO a `bing_linkedin_cache`. Postup by sme od tohto miesta mohli popísať nasledovne:

1. zadáme *„modules search“* a vyhľadáme potrebný modul
2. *„modules load názov-krok1“*
3. zadáme *„info“* pre zobrazenie potrieb modulu
4. *„options set SOURCE nazov“* - názov spoločnosti alebo domény (firma, firma.eu/cz/net,...)
5. nakoniec spustíme pomocou *„run“*
6. po preskúmaní všetkých plánovaných zdrojov zadáme *„back“*, zmeníme modul a zopakujeme postup

Výsledky sa automaticky ukladajú do databázy používaného workspacu. Po vykonaní všetkých modulov pre nami požadované domény a názvy môžme zobraziť výsledky pomocou *„dashboard“* a konkrétne výstupy pomocou príkazu *„show“*, konkrétne *„show hosts/domains/profiles,...“*.

rowid	host	ip_address	country	module
1	example.domena.cz		Czech Republic	bing_domain_web
2	example.domena.net		Czech Republic	bing_domain_web
3	example2.store.cz	192.168.52.90	Czech Republic	certificate_transparency

Tabuľka 1: Príklad výpisu Recon-ng

Príklad výpisu workspacu v tabuľke 1, nám znázorňuje tvar získaných dát. Z nich môžme vyčítať danú subdoménu, použitý modul a pri iných moduloch aj ostatné parametre ako IP adresa a čísla otvorených portov. Tieto parametre nie sú zobrazené, je uvedený iba príklad, z dôvodu citlivosti týchto dát.
Pri zobrazení položiek profiles alebo contacts môžme vyčítať zistené informácie o vyhľadanom objekte ako: registrácia na internetových stránkach (účty na sociálnych sieťach, streamovacie služby, spravodajské denníky, online hry a mnoho ďalších možností), mená a pozícia zamestnancov (Linkedin, vyhľadávanie Bing), emailové adresy. Tieto údaje môžu byť duplicitné alebo neúplné preto je nutná filtrácia avšak poskytnú nám množstvo informácií ktoré môžme neskôr použiť.

3.2.1.2 API kľúče alebo API keys sú kľúče ktoré sú vyžadované niektorými modulmi nástroja recon-ng k svojej funkcionalite. Tieto kľúče nám umožnia priami prístup do rôznych aplikácií ako Linkedin, Bing, Twitter, Github a iné. API kľúče treba uchovávať v súkromí keďže ide o, súkromne vlastnený, prístupoví kľúč registrovaného účtu do aplikácie. Postup pre získanie kľúča:

- registrácia na danej stránke (príklad vyhľadávanie Bing)
- vyhľadanie API kľúča v nastaveniach prihláseného účtu
- pridanie kľúča do recon-ng pomocou príkazu „ `keys add bing_api API_KEY "`
- zobrazenie všetkých modulov s prípadne pridanými kľúčmi pomocou „ `keys list "`

3.2.2 Nástroj Dmitry

Dmitry je program vytvorený za účelom zhromaždenia čo najviac informácií o hostovy v čo najkratšom čase. Jeho základná funkcionalita umožňuje zhromažďovanie informácií o cieľovom hostiteľovi od jednoduchého vyhľadávania na cieľovom serveri až po hlásenia UpTime a scan TCP portov. [10]

Použitie Dmitry je jednoduché. Celá aplikácia obsahuje 10 parametrov.

- -o uloží výstup ako .txt súbor

- -i Vykoná whois lookup IP adresy hosta
- -w Vykoná whois lookup domény hosta
- -n Vráti Netcraft.com informácie o hostovy
- -s Vyhľadá prípadné subdomény
- -e Vyhľadá prípadné emailové adresy
- -p Vykoná TCP port scan na hostovy
- -f Vykoná TCP port scan na hostovy a vypíše aj filtrované porty
- -b Prečíta banner scanovaného portu
- -t Nastaví TTL pri scanovaní TCP portov na 0-9 sekúnd

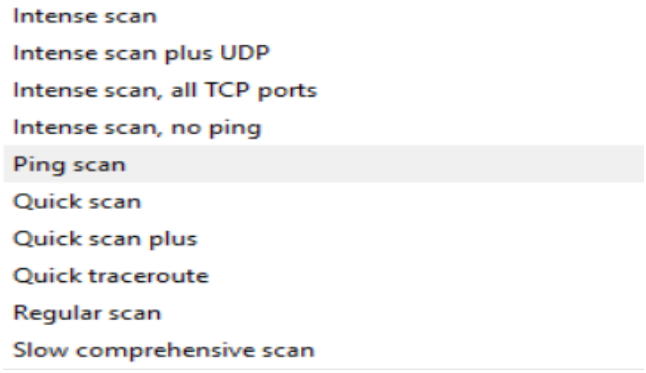
Spustíme Dmitry a zadáme „ `-winsepfb hladana_domena.eu "`.

3.2.3 Nástroj (Ze)Nmap

Nmap, alebo jeho GUI verzia Zenmap, je všestranný open-source nástroj. Je to veľmi silný nástroj používaný na bezpečnostný audit alebo mapovanie sietí. Dokáže vyhľadať hosta, služby, operačné systémy, paketové filtre / brány firewall a mnoho ďalších charakteristík.[7] Nmap dokáže ľahko skenovať veľké siete ale iba jedného hosta. Obsahuje množstvo parametrov a ich možných kombinácií a nie je možné sa im venovať všetkým. V tejto časti si ukážeme príklad práce so Zenmap v prostredí Windows.

3.2.4 Práca v (Ze)Nmap

Na začiatku musí byť definovaný **Target** teda náš cieľ, či už vo forme domény (priklad.scan.net) alebo IP adresy. Ďalším krokom je zadanie parametrov príkazu. Zenmap ponúka niekoľko prednastavených scanov.



Obr. 1: Zenmap prednastavené typy scanu

Tieto možnosti sú len rôznymi, často používanými, kombináciami rôznych parametrov. Predtým ako prejdeme k jednotlivým príkazom je dôležité si definovať spôsob scanovania. Prvým a najdôležitejším bodom je mať súhlas, najlepšie písomný, klienta so scanovaním domény/serveru/webu. Ďalším bodom je spôsob prístupu a teda či ide o otvorený bezpečnostný audit a scan môže prebiehať voľne. Alebo ide o bezpečnostný test a scan má byť čo najmenej, najlepšie vôbec, detekovateľný klientom.

V prípade pridelenia domény, napríklad test.domena.net, môžme postupovať nasledovne: " `nmap -sV -T4 -6 -O -F --version-light test.domena.net` " kde

- nmap - príkaz spustenia nástroja nmap
- -sV - testuje otvorené porty na určenie služby/verzie
- -T4 - časoví interval <0,5> medzi jednotlivými odoslanými požiadavkami kde 0 je najpomalší, 5 môže zahltiť systém v závislosti na jeho výkone
- -6 - povoliť scan pre IPv6
- -O - detekuje operačný systém
- -F - vykonať rýchli scan len definovaných portov (vnútorne definovaných v nástroji Nmap)
- –version-light - Rýchla identifikácie verzie, vykonanie len najviac pravdepodobných testov na úspech
- doména test.domena.net

V prípade pridelenia rozsahu IP adresy, zameníme doménu za adresu a pokračujeme. V prípade pridelenia bloku adries, príklad 8.8.8.0/24, môžme použiť niekoľko príkazov a to:
" `nmap 8.8.8.0/24 ...` "
" `nmap 8.8.8.1-14 ...` "
" `nmap 8.8.8.* ...` " pre 8.8.8.1 - 8.8.8.256
" `nmap 8.8.8.* --exclude 8.8.8.1 ...` " pre všetky okrem 8.8.8.1

Tento základný ale nápadný scan dokáže zistiť IP adresu, stav prvých 100 najpoužívanejších portov na nich bežiace služby, prípadne verzie a ich stav (open/closed), typ jadra kernelu, a nakoniec operačný systém s jeho generáciou na základe vyhodnotenia odpovedí vo forme pravdepodobnosti. Tento príkaz je výborným odrazovým mostíkom pri zbere informácií.

3.2.4.1 Maskovanie identity V prípade skrytého bezpečnostného auditu je nutné svoju aktivitu skrývať poprípade maskovať. Nmap ponúka niekoľko možností ako tento cieľ dosiahnuť pomocou pár príkazov. Ako vzor použijeme predošlí scan:

„ nmap -sV -T2 -6 -O -F –version-light –spoof-mac 0 -S 192.168.54.78 -D 68.54.78.52, 68.53.72.8,168.172.68.20,155.192.92.85 test.domena.net “

Pridané parametre:

- –spoof-mac 0 - zmení MAC adresu na inú náhodne generovanú
- -S - zmení IP adresu na vloženú "falošnú"
- -D - Decoy alebo návnada, bude posielať falošné žiadosti zo zadaných IP adries, pri väčšom počte môžu mať stanice problém identifikovať pravú adresu avšak nmap môže mať problém s generovaním odpovede

Úroveň zabezpečenia identity záleží od konfigurácie serverového firewallu a schopností administrátora. Pre úplnú anonymitu je nutné použiť ďalšie maskovacie prvky (VPN, proxy server,...).

Nmap obsahuje enormné množstvo parametrov, prístupov a možností práce so skriptami, kvôli tomu nie je možné ho obsiahnuť detailne. Príklady niektorých často používaných príkazov a skriptov sa tak nachádzajú v prílohe. Jednotlivé parametre a funkcie Nmapu sa nachádzajú na oficiálnych stránkach [7].

3.2.4.2 Získavanie HTTP záhlavia

kde HTTP záhlavie sa môže klasifikovať na 2 hlavné časti a to HTTP žiadosť a HTTP odpoveď. Prehliadač klienta odošle žiadosť HTTP so záhlavím obsahujúcim údaje o prehliadači ako typ, verzia, jeho schopnosti, typ operačného systému klienta a typ prijímaných odpovedí. Server na základe žiadosti odpovie HTTP odpoveďou ktorá obsahuje podrobnosti o servery ako typ, používané kódovanie, cookies a mnoho iných informácií. Odchytávanie HTTP záhlavia poskytuje informácie o priebehu komunikácie klient-server. Tieto informácie sú pre znalého útočníka dôležité, môžu byť zmanipulované alebo použité pri príprave vektorov útoku. [6]

Príklady http odozvy s použitím zabudovaných skriptov:

nmap –script=http-title 188.210.180.205

```
80/tcp  open  http

| http-title: GQS - Responsive Bootstrap 4 Admin Dashboard

|_Requested resource was http://test.domena.net/Identity/Account/Login?
    ReturnUrl=%2F
```

nmap –script=http-headers 188.210.180.205

nmap –script=http-enum 188.210.180.205

Vo výpise title, Nmap podľa Ip adresy zistil nastavený port, jeho stav a doménu ktorá je na ňom spustená. Podľa aktuálnej URL môžme zistiť že ide o prihlasovaciu stránku. Podľa výpisu headers ktorý sa nachádza v prílohe C, dokážeme vyčítať typ kódovania, druh obsahu, cookies, typ serveru, pridelenú identitu, použitý nástroj na vývoj a mnoho iných informácií slúžiacich k naštudovaniu cieľa a možných prienikov jeho serveru.

3.2.5 Nástroj TheHarvester

Cieľom tohto programu je zhromažďovať e-maily, subdomény, hostov, mená zamestnancov, otvorené porty a bannery z rôznych verejných zdrojov a porozumieť internetovej stope klienta. Je to užitočné pre každého, kto chce vedieť, čo môže útočník vidieť o svojej organizácii. [8]

TheHarvester je jednoduchý a výkonný nástroj. Pri používaní musíme dbať na použité parametre z dôvodu oddelenia pasívnych a aktívnych prvkov. Ak chceme vykonávať skrytý scan nemali by sme používať aktívne prvky. a tak by sme mohli rozdeliť použitie na 2 príkazy.

pre pasívny scan: **theharvester -d test.domena.eu -l 500 -b all -s -v**

Kde -d udáva hľadanú doménu, -l maximálny počet výsledkov na vyhľadávanie, -b všetky zdroje (alebo môžme konkretizovať) ako google, baidu, linkedin, bing, yahoo, twitter a mnoho ďalších. Parameter -s je vyhľadávanie Shodan ku ktorému je nutné pridať API kľúč. Posledné -v vyhľadá virtuálnych hostov za pomoci Bing vyhľadávania.
pre aktívny scan použijeme: **theharvester -d test.domena.eu -l 500 -b all -p -t -n -c**

-p scanuje a detekuje hostov na portoch 21,22,80,443,8080. Ďalšie parametre sa zaoberajú prácou s DNS konkrétne -t TLD vyhľadávanie, -n DNS lookup a -c brute force útok na doménu.

S použitím týchto nástrojov sme získali množstvo, potrebných aj redundantných, informácií. Príkladom sú domény a subdomény spolu s ich IP adresami, mená potencionálnych zamestnancov a mailové adresy, stav portov, serverové informácie ako aj základný prehľad o HTTP komunikácií. Je na testerovy tieto informácie prefiltrovať, zhodnotiť a vypracovať svoju mienku o prístupe k tomuto testu. A to je ďalším krokom tejto práce.

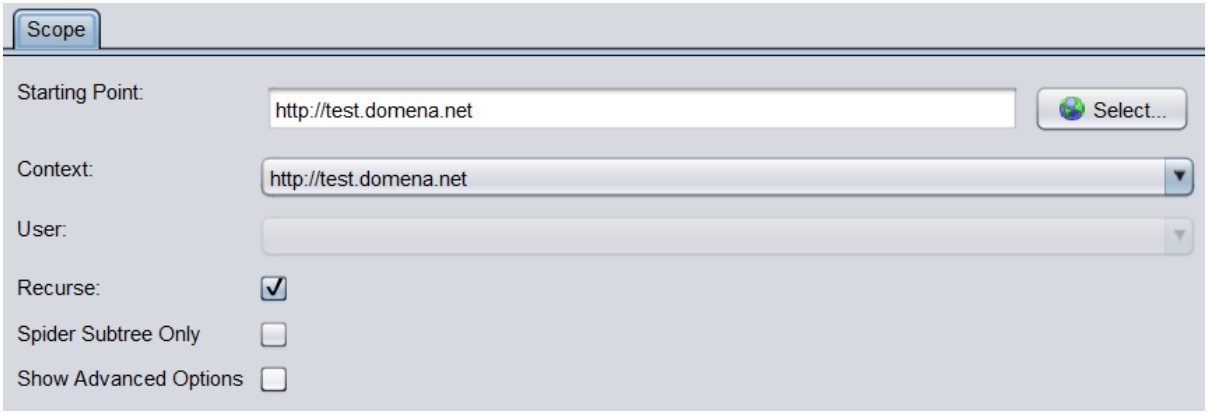
3.3 Posudzovanie zraniteľností

Ako už bolo povedané v sekcií 2.5, táto časť je fázou aktívneho posudzovania zraniteľností. Niektoré scany vykonávané v sekcií 3.2 by sa istým spôsobom dali považovať za aktívne. V tejto podkapitole ale začnem s používaním nástroja OWASP ZAP, ktorý dokáže zraniteľnosti vyhľadať a vyhodnotiť automatizovane. Po automatizovanej časti nasleduje časť manuálneho overovania,

vzhľadom na to, že automatizované testovanie, nie je schopné intuitívneho uvažovania, a môže prehliadnuť prípadné zraniteľosti.

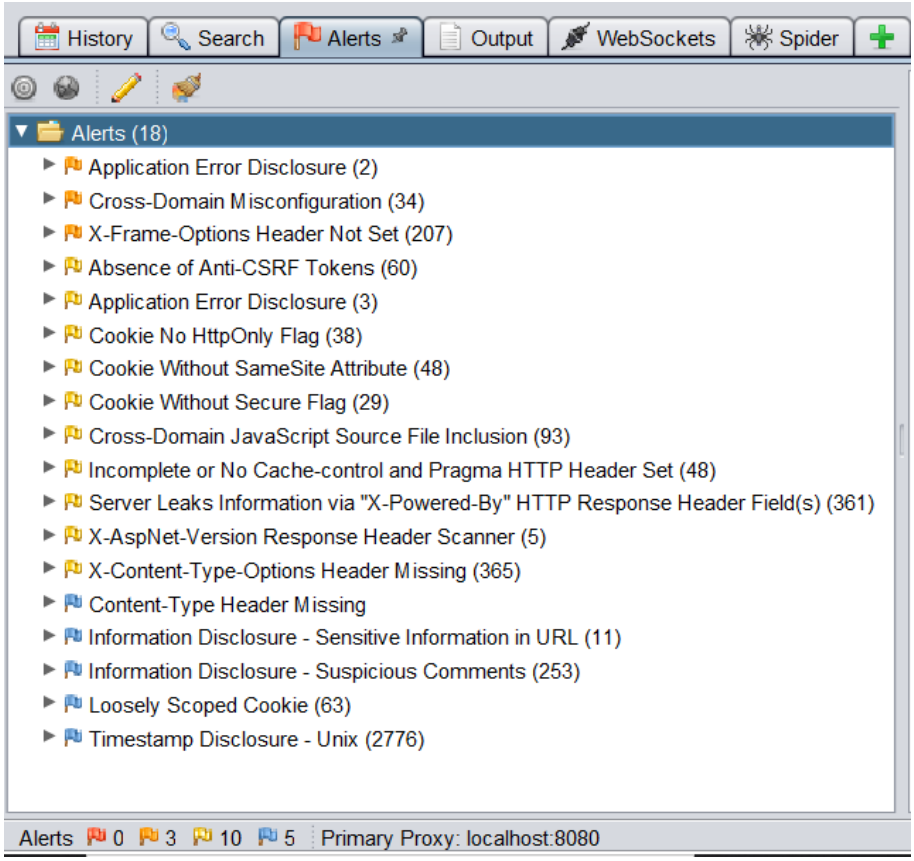
3.3.1 Určenie rozsahu a prehľadávanie adresárov

Prvým krokom je spustenie aplikácie OWASP ZAP. Zapneme prehliadač a zadáme cieľovú URL. Vďaka tomu, že OWASP ZAP sa chová ako proxy, môžme vidieť zobrazené stránky a ich štruktúry. Avšak vidíme všetky data, prechádzajúce skrz proxy, a to je neprehľadné. Aby sme sa zamerali len na náš cieľ musíme určite rozsah - scope. Pravým tlačítkom klikneme na najvyššiu doménu, vyberieme Ïnclude in Context"vyberieme "New Context"dáme OK. V okne sa vytvoril nový kontext s názvom stránky. Vyberieme ho a zaklikneme "Show only URLs in Scope". Tak docielime aby sme videli a pracovali len s tým čo potrebujeme. Klikneme na doménu pomocou RMB a vyberieme Ättackä "Spider". Rekurzia nám zabezpečí, že sa program bude snažiť prehľadávať URL rekurzívne čo najviac do hĺbky alebo do maximálne nastaveného obmedzenia.



Obr. 2: OWASP ZAP - spider

Na úvod sa zdá, že prehľadávanie prebehlo úspešne, avšak bolo zastavené prihlasovacím systémom. Tuto prekážku obídeme použitím poskytnutých prihlasovacích údajov. Z plochy vyberieme "Manual Explore", zadáme URL a ako prehliadač sa zvolí Chrome. V prihlasovacom okne vyplníme poskytnuté údaje a prihlásime sa. Prihlásením sa vytvoril token. Vrátime sa do programu OWASP ZAP a znovu spustíme "Spider". Tento krát už prehliadne celú aplikáciu čo sa odrazí na čase scanu a počte nájdených URL. Na karte Älerts"sa nachádza zoznam všetkých nájdených podozrení ktoré by sa mohli dať zneužiť.



Obr. 3: OWASP ZAP - Výsledky Spider scanu

Tento výsledok automatizovaného scanu je veľmi informatívny. Ako bolo však povedané je potrebné vykonať manuálny posudok, to znamená ručne prejsť stránku a vyhodnotiť možnosti útokov, skontrolovať validácie polí, chybové hlášky a ostatné informatívne prvky, ktoré scan nemusel zachytiť.

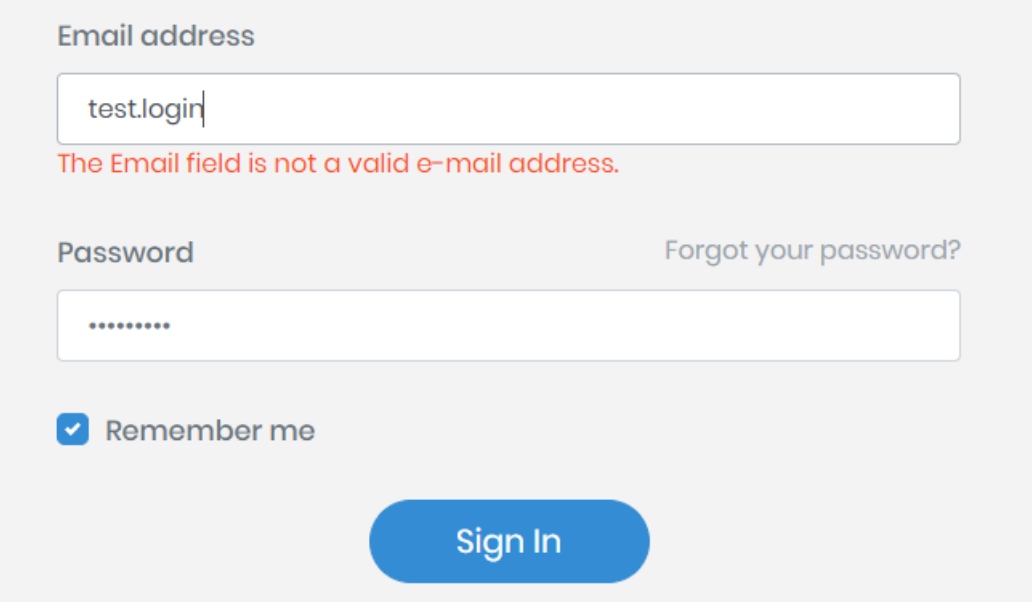
3.3.2 Manuálne posudzovanie zraniteľností

Manuálne posudzovanie zraniteľností je "ručné"prechádzanie stránky. Pri tomto kroku musí tester, k posúdeniu situácie, využiť svojich znalostí a skúseností. Najlepším príkladom manuálneho testu sú polia, validácie a dizajn aplikácie.

Všeobecný postup testovania polí v tejto práci je nasledovný:

- vloženie validných údajov a pozorovanie chovania
- vloženie nesprávnych údajov a vyhodnotenie chovania, prípadne chybových správ
- vloženie špeciálnych znakov
- vloženie JavaScriptu
- vloženie SQL Injekcia

3.3.2.1 Chybné validácie formulára sú príkladom nesprávnej validácie a dizajnu. Takúto chybu automatizovaný scan nie je schopný odhaliť. Príkladom v tejto aplikácií je prihlasovací formulár.



Obr. 4: Nesprávny prihlasovací formulár

Údaje a chybové správy každého formulára, by mali mať neutrálny charakter. To znamená, že osoba bez znalosti logiky aplikácie by nemala byť schopná zistiť viac ako jej je poskytnuté. Avšak ako môžme vidieť na obrázku 4, prihlasovacím menom je emailová adresa. Z takejto informácie dokáže útočník odhadnúť prihlasovacie meno a to vďaka zberu informácií v sekcií 3.2. Útočník mohol nájsť mená zamestnancov a ich emailové adresy, alebo z mien vytvoriť rôzne varianty adries. Ďalším zistením je neprítomnosť akéhokoľvek ochranného mechanizmu po opakovanom vložení nesprávnych údajov. Z týchto zistení vyplýva záver, že prihlasovacia stránka je náchylná na útok "hrubou silouälebo slovníkový útok.

3.3.2.2 SQL injekcia je ďalším dobrým príkladom manuálneho posudzovania zraniteľností. Je to typ útoku, ktorým je útočník schopný ovplyvňovať požiadavky aplikácie na databázu. Správna manipulácia umožní zobraziť dáta alebo informácie o architektúre aplikácie ktoré by, za normálnych okolností, nemali byť zverejnené. Z kapitoly zberu informácií, za pomoci programu Nmap 3.2.4, a výpisu HTTP hlavičky v prílohe C môžme vyčítať, že aplikácia používa jazyk ASP.NET. Tento jazyk, je pri použití databázového systému SQL Server, náchylný na tento typ útoku. Jediné čo ostáva je túto možnosť otestovať.

Prvým krokom je výber vhodného poľa. V prípade tejto aplikácie, je to akýkoľvek vyhľadávací filter s použitím reťazca znakov. Do poľa zadáme ľubovoľný, nedeštruktívny SQL príkaz ako napríklad **SELECT @@version**, na zobrazenie verzie databázy. Vykonanie príkazu nám

nevrátilo žiadny výsledok, avšak príkaz bol spracovaný. SQL injekcia je možná ak sa vložený príkaz spracuje, s vrátením výsledku či bez, alebo vyvolá chybu na strane servera či stránky. Injekcia nie je možná iba v prípade ak sa vyvolá chybová, validačná správa. V tomto prípade po spracovaní našej injekcie, môžme predpokladať jej funkčnosť. Samozrejme čím väčšia pestrosť príkazov, tým presnejší vieme urobiť odhad. Avšak SQL injekcia ako taká je mimo rozsahu tejto práce.

Po týchto ukážkach, scanoch a testoch, môžme prejsť na časť reálnej snahy zneužitia systému, za pomoci nájdených zraniteľností.

3.4 Pokus o zneužitie zraniteľností systému

V tejto fáze sa útočník snaží o prienik alebo zneužitie systému. Tieto kroky sú vykonané na základe informácií získaných z predošlých fázy testovania. Podľa zistených informácií, môžme začať logicky öd prednej strany", a to testovaním prihlasovacieho formulára.

3.4.1 Prihlasovací formulár

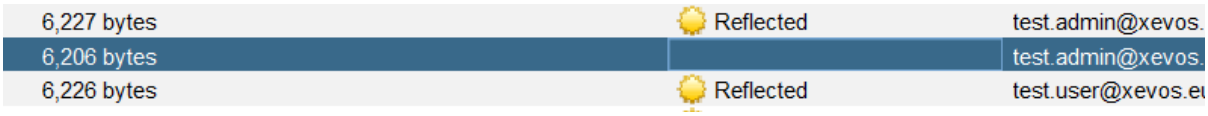
V sekcií 3.3.2 boli uvedené problémy s prihlasovacím formulárom ako napríklad neprítomnosť obranných mechanizmov pri vkladaní údajov. To robí formulár náchylní na útoky hrubou silou, alebo slovníkové útoky. Taktiež je tu možnosť SQL injekcie, ale odhadom veľmi nepravdepodobná. A tak prejdeme k praktickému výkonu slovníkového útoku za pomoci OWASP ZAP.

3.4.1.1 Pokus o prienik prihlasovacím formulárom uskutočňujeme pomocou aplikácie OWASP ZAP alebo iného vhodného prostredia na tento účel. Po správnej konfigurácií prejdeme na *„Manual Explore“* kde zadáme URL webu a ako prehliadač zvolíme Chrome. Po spustení prehliadania zistíme, že sa nachádzame na prihlasovacom formulári. OWASP ZAP nám ako proxy odchytáva všetku komunikáciu či už prichádzajúcu alebo odchádzajúcu. Tejto vlastnosti môžme využiť, a to tak, že budeme manipulovať dátami požiadavkov. Z toho vyplýva, že prvým krokom odchytenia požiadavku, je požiadavku vytvoriť. To vykonáme pomocou náhodného pokusu o prihlásenie. Do poľa Email zadáme náhodnú emailovú adresu ako napríklad „email.test@niekto.cz“, a do hesla zadáme reťazec znakov. Reťazec hesla by mal, najlepšie, obsahovať veľké písmena, čísla a špeciálne znaky, a to preto aby sme zistili prípadne šifrovanie komunikácie. Ako heslo zadáme „ HesLo78+-* “ a prihlásime sa. Prihlásenie bolo, samozrejme, neúspešné. Vo filtri odchytíme POST požiadavku na server s prihlásením. Po otvorení požiadavku sa dostaneme k jeho HTTP hlavičke a telu. V tomto prípade je zaujímavejšou časťou telo požiadavku.

```
Input.Email=email.test%40niekto.cz&Input.Password=HesLo78%2B-*+&Input.
    RememberMe=true&
```

```
__RequestVerificationToken=
    CfDJ8JPU4TOkOq5Mjc8Z79iKT5m05etZiHWdI8qUoYMOwy4vRsd3Vrfq
&Input.RememberMe=false
```

Vo výpise vidíme náš vstup na email, heslo, overovací token serveru a možnosť uloženia prihlásenia. Ďalej vidíme, že niektoré znaky majú špeciálne kódovanie. Ide o URL kódovanie. Na útok hrubou silou použijeme funkciu *Fuzzer*. Označíme hodnotu emailu a po stlačení RMB vyberieme funkciu *„Fuzz“*. Klikneme na *„Payloads“* kde môžme vložiť naše dáta. Ide o emailové adresy ktoré sme lokalizovali v sekcií 3.2. Najpraktickejším riešením je vloženie súboru s týmito adresami. Tento súbor musí byť pred vložený kódovaný na URL, čo vykonáme za pomoci rôznych online konvertorov dostupných na internete. Vyberieme cestu k súboru a dáme *„OK“*. Po vložení súboru z adresami nasledujú heslá. Označíme vloženú hodnotu hesla a klikneme na *„Add“* čo nám zobrazí *„Payloads“*, a znovu pridáme URL kódovaný súbor z heslami. Súbory z heslami sú voľne dostupné na internete, sú súčasťou systému Kali Linux alebo si každý môže vytvoriť svoj vlastný. Dostupné zbierky hesiel sú zoskupením uniknutých, najčastejšie používaných hesiel na svete. Po nastavení oboch parametrov prejdeme na karte na možnosť *„Options“*, kde zaklikneme pole *„Follow Redirects“*. Overovací token serveru riešiť nemusíme, pretože ten si zisťuje a posiela OWASP ZAP automaticky pokiaľ mu nepovieme inak. Po tomto nastavení zaklikneme *„Start Fuzzer“* a počkáme na výsledky.



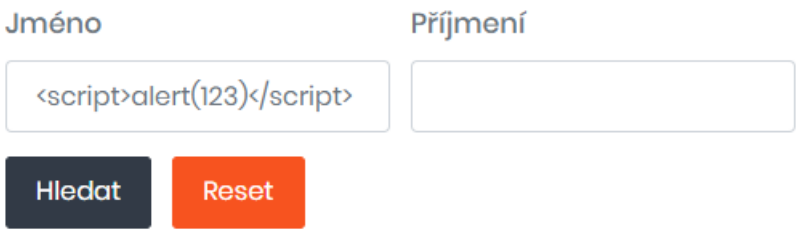
Obr. 5: Prelomenie kombinácie mena a hesla

Po vykonaní útoku hrubou silou musíme vedieť, kde a čo hľadať vo veľkom množstve výsledkov. Najčastejším znakom pozitívneho výsledku sú rôzna veľkosť tela odpovede alebo hlavičky od ostatných, vyšší čas omeškania alebo, pokiaľ sme nastavili „Follow Redirects“, absencia príznaku „Reflected“. Otvoríme požiadavku pri ktorej si myslime, že mohla byť úspešná, vyberieme z nej prihlasovacie údaje ktoré použila, a tie použijeme k prihláseniu.

3.4.2 Cross-Site skriptovanie

Vzhľadom na vysoký počet tabuliek, filtrov a vstupov webovej aplikácie je správne predpokladať možný úspech útokov XSS. K tým dochádza, keď útočník využíva webovú aplikáciu na odoslanie škodlivého kódu, zvyčajne vo forme skriptu na strane prehliadača, inému koncovému používateľovi. Tento vstup následné generuje výstup, bez toho, aby bol rázne validovaný alebo kódovaný. Tento typ útoku je veľmi bežný preto je nutné ho za každých okolností otestovať.

3.4.2.1 Testovanie Reflektívnych XSS útokov prebieha pomocou vkladania skriptov do polí vstupov a analýzy výstupu. Ako vstup som si vybral jeden z filtrov vyhľadávania. Zadáme jednoduchý skript na otestovanie validácie a to `<script>alert(123)</script>`.



Obr. 6: Jednoduchý test skriptu

Po vyhľadaný sa skript vykoná a odpozorujeme chovanie stránky. Tabuľka takýto vstup vôbec neočakávala. Nastala chyba a vypísal sa interný statický kód tabuľky zobrazovania dát. Tento kód uvádza informácie o metóde, názvoch parametrov kódu a dokonca kus JavaScriptového kódu. Takéto interné informácie sú veľmi citlivé a môžu napomôcť k presnejšiemu nasmerovaniu útokov. Ako ďalší príkaz zadáme:

javascript:/–></title></style></textarea></script></xmp><svg/onload='+/"/+ /onmouseover=1/+/[*/[]/+alert(1)//'>*

Tento, verejne dostupný, príkaz vykoná niekoľko kontextov ako html, javascript, skript reťazec a URL, naraz. Po vykonaní sa skript vykoná, zobrazí sa chcený výsledok a tabuľka znovu chybuje a zobrazuje svoj JavaScriptový kód. Ďalším krokom sú testovania rôznych príkazov, ich variácií s úvodzovkami alebo bez a manipulácia rôznych znakov. Použité príkazy sú súčasťou databáze príkazov nadácie OWASP [11]. Jednoduchým testovaním sme zistili, že stránky pri konfrontácií s XSS útokom zverejňuje citlivé dáta statického kódu a v niektorých prípadoch sa dostane do chybového stavu. Týmto testom sme odhalili nepripravenosť systému na útoky reflektívnym XSS.

3.4.2.2 Testovanie DOM XSS kde DOM je štrukturovaný formát používaný na zobrazenie dokumentov v prehliadači. DOM umožňuje skriptom odkazovať na komponenty dokumentu ako pole alebo cookies. K tejto zraniteľnosti môže dojsť, keď je aktívny obsah, ako napríklad JavaScript, modifikovaný externou požiadavkou tak, že prvok DOM sa stane ovládateľný útočníkom. DOM útoky nemusia, a pravdepodobne nebudú, detekovateľné bežnými XSS filtrami.

Prvým krokom testovania DOM útokov je voľba vhodného poľa. Vzhľadom na štruktúru DOM je vhodné vybrať formulár, ktorý bude posielať žiadosť na server. Zvolíme predo funkciu vytvorenia nového objektu, alebo úpravy objektu. Zvolíme upraviť existujúci objekt a zmeníme

jeden z parametrov, pomocou nasledujúceho DOM kódu:

```
<script>
document.write("Stranka sa nachadza na: " + document.location.href + ".");
</script>
```

Po uložení zmien, nás stránka automaticky vráti na tabuľku ktorá nám dáta zobrazuje. Skript bol vykonaný a dokument stránky úspešne prepísaný ako vidíme vo výpise.

```
Stranka sa nachadza na: http://test.domena.net/People
```

Zistili sme, že DOM skripty sa dajú bezproblémovo vkladať do polí objektov a tak modifikovať dokument webového serveru. Po vložení škodlivého skriptu môže útočník manipulovať, alebo získať, dáta každého užívateľa, ktorý si zobrazí stránku na ktorej je skript vložený.

Po skončení útočnej fázy testovania, je nutné využité aj nevyužité zraniteľnosti analyzovať, vyhodnotiť a odporučiť prípadné zmeny systému k ich odstráneniu.

3.5 Analýza zraniteľností a ich vyhodnotenie

V tejto časti práce sa analyzujú nájdené zraniteľnosti spolu s výpisom možných riešení na ich opravu. Zraniteľnosti boli identifikované v sekcií 3.3 a jej podkapitolách.

3.5.1 Absencia ochranných mechanizmov prihlasovacieho formulára

V prihlasovacom formulári absentuje ochrana opakovaného prihlasovania. Prihlasovacím menom je emailová adresa, tento údaj nám prezrádza samotný formulár. Je možné pokúšať sa o prihlásenie v neobmedzenom počte, čo zanecháva aplikáciu zraniteľnú voči rôznym útokom prelamovania hesiel a útokom hrubej sily.

Úroveň rizika: Vysoká

Odporúčaný postup riešenia:

1. Zmeniť názov poľa prihlasovacieho mena z Emailová adresa na neutrálnu formu, napríklad Prihlasovacie meno, Názov účtu a iné.
2. Zmeniť validačné správy na neutrálnejšiu, menej špecifickú formu.
3. Zaviesť bezpečnostnú komponentu prihlasovania ako Captcha alebo odoprenie prihlasovania na určitý časový interval po opakovanom vložení nesprávnych údajov.
4. Ak je možnosť, použiť 2-krokové overenie prihlasovania.

3.5.2 Zraniteľnosť na XSS útoky

Stránka je vysoko nechránená proti XSS útokom typu reflektívneho a DOM. Vložené XSS potvrdené neboli. Škodlivý skript má prístup k akýmkoľvek súborom cookie, tokenom relácií alebo iným citlivým informáciám, ktoré si prehliadač ponechal a ktoré sa používajú na tomto webe. Tieto skripty môžu dokonca prepísať obsah stránky HTML.

Úroveň rizika: Vysoká

Odporúčaný postup riešenia:

1. Filtrácia vstupu v okamihu prijatia od používateľa. Filtrácia by mala prebiehať čo najpresnejšie na základe očakávaných dát poľa. Príklad: použitie bieleho prístupového listu (white list).
2. Kódovanie údajov na výstupe. V okamihu, keď sú užívateľsky ovládateľné údaje na výstupe v odpovediach HTTP, zakódujte výstup, tak aby sa zabránilo interpretácii výstupu ako aktívneho obsahu.
3. Použite vhodné hlavičky odpovedí. V odpovediach HTTP, ktoré nie sú určené na to, aby obsahovali HTML alebo JavaScript, použiť hlavičky Content-Type a X-Content-Type-Options.
4. V prípade nutnosti použiť politiku zabezpečenia obsahu. A teda kontrolu obsahu posielaného v aplikácií.

3.5.3 SQL injekcia

Priama zraniteľnosť na tento typ útoku nebol potvrdený. Databáza priamo nevrátila žiadne dáta. Avšak aplikácia priala a poslala všetky vložené injekcie. Zneužitie,tejto chyby, môže potencionálne viesť k získaniu citlivých údajov, ako napr. prihlasovacie údaje, osobné údaje. V niektorých prípadoch môže viesť aj k vykonaniu systémového príkazu, prípadne k ovládnutiu celého serveru.

Úroveň rizika: Stredná

Odporúčaný postup riešenia:

1. Použitie pripravených výkazov (parametrizované vyhľadávanie)
2. Overovanie užívateľského vstupu
3. Použitie uložených postupov

4. Poskytovať užívateľom prednastavené vstupy

5. Prideľovať len najnutnejšie privilégiá

3.5.4 Zverejnenie chyby aplikácie

Stránka obsahuje chybové hlásenie, ktoré môže zverejňovať citlivé informácie, ako napríklad umiestnenie súboru, ktorý spôsobil neošetrenú výnimku. Tieto informácie môžu byť použité na spustenie ďalších útokov proti webovej aplikácii.

```
throw new Error("iconv-lite internal error: invalid decoding table value " +
    uCode + " at " + nodeIdx + "/" + curByte);
```

Úroveň rizika: Stredná

Odporúčaný postup riešenia: Skontrolovať zdrojový kód tejto stránky. Implementovať vlastné chybové hlásenia. Zvážiť implementáciu mechanizmu, ktorý klientovi (prehliadaču) poskytoval jedinečný odkaz na chybu.

3.5.5 Nesprávna konfigurácia Cross-Domain

Z dôvodu nesprávnej konfigurácie zdieľania zdrojov (CORS) na webovom serveri, je možné načítanie dát z webového prehliadača. Táto konfigurácia povoľuje doménam tretích strán používanie neoverených API. Túto nesprávnu konfiguráciu by mohol útočník použiť na prístup k údajom, ktoré sú dostupné neautentizovaným spôsobom.

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Age: 28528677
Cache-Control: public,max-age=31536000
Content-Type: application/javascript
Date: Fri, 08 May 2020 18:40:20 GMT
Etag: "801eb2228ad31:0+ident"
Last-Modified: Mon, 31 Jul 2017 18:09:21 GMT
Server: ECAcc (via/F380)
Timing-Allow-Origin: *
Vary: Accept-Encoding
X-Cache: HIT
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Length: 23261
```

Obr. 7: Cross-Domain Misconfiguration: HTTP hlavička

Úroveň rizika: Stredná

Odporúčaný postup riešenia: Zabezpečiť, aby citlivé údaje neboli dostupné neautentizovaným spôsobom (napríklad pomocou bieleho zoznamu adries IP). Nakonfigurovať HTTP hlavičku „Access-Control-Allow-Origin“ na reštriktívnejšiu skupinu domén alebo úplne odstrániť všetky hlavičky CORS.

3.5.6 Chýbajúce nastavenie v záhlaví

X-Frame-Options hlavička nie je zahrnutá v HTTP odpovedi na ochranu pred „ClickJacking“ útokmi.

Úroveň rizika: Stredná

Odporúčaný postup riešenia: Nastaviť hlavičku protokolu X-Frame-Options HTTP na všetkých webových stránkach vrátených vašim serverom. Ak sa očakáva, že stránka bude zarámovaná iba na stránkach vlastného serveru, použiť parameter SAMEORIGIN. Ak sa očakáva, že stránka by nemala byť zarámovaná nikdy, použiť parameter DENY. Príklad X-Frame-Options: DENY.

3.5.7 Absencia anti-CSRF tokenov

V odosielanom formulári HTML sa nenašli žiadne anti-CSRF tokeny. V CSRF ide o falšovanie HTTP žiadostí, bez vedomia užívateľa. Základnou príčinou je funkčnosť aplikácie, ktorá opakovateľným spôsobom využíva predvídateľné akcie formulárov. Charakter útoku spočíva v tom, že CSRF využíva dôveru, ktorú má webová stránka pre používateľa.

Úroveň rizika: Nízka

Odporúčaný postup riešenia:

1. Použitie knižnice alebo štruktúry ktorá nedovoľuje túto zraniteľnosť.
2. Pre každý formulár vytvoriť jedinečné jednorázové číslo a po prijatí formulára ho overiť.
3. Identifikovať obzvlášť citlivé operácie. Pri výkone takýchto operácií, odoslať samostatnú požiadavku, aby sme sa uistili, že užívateľ zamýšlal vykonať túto operáciu.
4. Nepoužívajte metódu GET na žiadnu požiadavku, ktorá spôsobuje zmenu stavu.
5. Skontrolovať hlavičku sprostredkovateľa HTTP, aby sa zistilo, či požiadavka pochádza z očakávanej stránky.

3.5.8 Nezabezpečený parameter cookie

Súbor cookie bol nastavený bez zabezpečeného príznaku, čo znamená, že k súboru cookie je možné pristupovať prostredníctvom nezašifrovaných spojení.

```
Set-Cookie: __cfduid=ded1cfc723e60dd7566eba64427256a981588963432; expires=Sun,
    07-Jun-20 18:43:52 GMT; path=/; domain=.datatables.net; HttpOnly; SameSite=
    Lax
```

Úroveň rizika: Nízka

Odporúčaný postup riešenia: Kedykoľvek súbor cookie obsahuje citlivé informácie alebo je tokenom relácie, mal by sa vždy odoslať pomocou šifrovaného kanála. Skontrolovať, či je pre súbory cookie obsahujúce takéto citlivé informácie nastavený zabezpečený príznak.

3.5.9 Cross-Domain inklúzia zdrojového súboru JavaScript

Stránka obsahuje jeden alebo viac súborov skriptov z domény tretej strany. Toto predstavuje riziko útoku pomocou skriptu, a tým spôsobiť nečakané zmeny stavov webovej aplikácie.

```
<script src="/lib/sweetalert2/sweetalert2.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/promise-polyfill@7.1.0/dist/promise.
    min.js"></script>
<script src="/js/alert-messages.js"></script>
```

Úroveň rizika: Nízka

Odporúčaný postup riešenia: Overiť, že zdrojové súbory JavaScriptu pochádzajú z dôveryhodných zdrojov. Pokiaľ ide o aplikáciu s obsahom veľmi citlivých dát, tak sa odkazovať jedine na svoj interný kód.

3.5.10 Neúplná alebo žiadna kontrola vyrovnávacej pamäte a nastavenia HTTP hlavičky

Riadenie vyrovnávacej pamäte a HTTP hlavička pragma neboli správne nastavené alebo chýbajú, čo prehliadaču a serverom proxy umožňuje ukladať obsah do vyrovnávacej pamäte.

```
Cache-Control: public, max-age=31536000
```

Úroveň rizika: Nízka

Odporúčaný postup riešenia: Ak je to možné, nastaviť hlavičku riadenia vyrovnávacej pamäte na: bez medzipamäte, bez ukladania, musí sa znovu validovať. Hlavičku HTTP pragma nastaviť bez vyrovnávacej pamäte. Príklad Cache-Control: no-cache.

3.5.11 Únik informácií serveru príčinou HTTP odpovedí

Serveru unikajú interné informácie skrz odpoveď HTTP. Webový server zverejňuje informácie prostredníctvom jedného alebo viacerých hlavičiek odpovedí HTTP typu „X-Powered-By“. Prístup k takýmto informáciám môže útočníkovy uľahčiť identifikáciu komponentov, na ktorých funguje webová aplikácia, a zraniteľnosti, ktorým môžu tieto komponenty podliehať.

```
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
```

Úroveň rizika: Nízka

Odporúčaný postup riešenia: Ubezpečiť sa, že webový alebo aplikačný server sú nakonfigurované na potlačenie týchto hlavičiek.

3.5.12 Chýbajúce nastavenie HTTP hlavičky

Nebola nastavená možnosť "X-Content-Type-Options"HTTP hlavičky na „nosniff“. To umožňuje starším verziám prehliadača, ako Internet Explorer aalebo Chrome, interpretovať telo odpovede v inom type ako bolo deklarované. To znamená, že sa nemusí zistiť skutočný typ odpovede, alebo sa môže zmanipulovať.

Úroveň rizika: Nízka

Odporúčaný postup riešenia: Nastaviť hlavičku ako: X-Content-Type-Options: nosniff.

Po vykonaní všetkých testov, analýzy zraniteľností a vypracovaní záverov, posledným krokom je tvorba dokumentácie. Dokumentácia postupov, pokusov a výstupov vytvorí konečnú správu (ang. report). Táto správa slúži ako výstup pre klienta k pochopeniu priebehu a výsledku testu. Dokumentovať sa môže celý proces, ako aj jednotlivé kroky. A týmto krokom sa pomaly približuje záver práce.

3.6 Dokumentácia

Ako už bolo povedané, poslednou, a dôležitou, časťou každého testovania je vytváranie reportu, či už čiastočného alebo úplného. V tejto časti sa nachádzajú ukážky práce s použitými nástrojmi k tvorbe týchto reportov.

3.6.1 Recon-ng

Generovanie reportov v nástroji recon-ng podporuje hneď niekoľko formátov a to:

- csv
- html
- json
- list
- proxifier
- pushpin
- xlsx
- xml

Postup generovania reportov:

1. vyberieme požadovaný workspace
2. spustíme požadovaný modul „`modules load reporting/format`", príklad: reporting/csv
3. zadáme „`info`" pre zobrazenie potrieb modulu
4. použijeme „`options set parameter hodnota_parametru`" a vyplníme parametre podľa potreby
5. spustíme pomocou „`run`"
6. po úspešnom vygenerovaní sa zobrazí cesta k report súboru

3.6.2 Dmitry

Dmitry používa ako možnosť výstupu parameter „ `-o nazov_suboru` ". Kompletný príkaz vyzerá nasledovne: „ `-winsepfb -o nazov_reportu hladana_domena.eu` ".

Dmitry používa k výstupu jedine .txt súbory!

3.6.3 (Ze)Nmap

Nmap výstup je možný v 5 formátoch. Report vytvoríme pridaním parametru do reťazca. Možnosti:

1. -oN <nazov> - vytvorí bežný výstup (prípona .nmap)

2. -oX <nazov> - XML výstup

3. -oS <nazov> - script kiddie výstup

4. -oG <nazov> - grepable výstup (zastaraný)

5. -oA <nazov> - ukladá výstup do XML, grepable a bežného naraz (3 samostatné súbory)

Zenmap ako GUI nám umožňuje uložiť získané data nasledovne v hornom menu:

„Scan -> Save scan -> Výber konkrétneho scanu -> Save -> Voľba úložiska -> Voľba formátu .xml/.nmap -> Save “

alebo pre všetky aktivne scany:

„Scan -> Save All Scans to Directory -> Voľba úložiska -> Save “

3.6.4 TheHarvester

Generovanie výstupu sa vykonáva za pomoci pridania parametru `„ -f nazovSúboru.prípona "` do vykonávaného reťazca. Výpis programu sa zapíše a uloží do vytvoreného súboru vo formátoch HTML alebo XML. Bez zadania prípony formátu súboru sa vytvoria oba formáty výstupu.

3.6.5 OWASP ZAP

Nástroj OWASP ZAP prichádza s veľmi jednoduchou generáciou reportov. Jediná požiadavka na užívateľa je otvoriť *„Report“* v lište nástrojov a zvoliť si požadovaný formát, ako:

1. HTML

2. XML

3. Markdown

4. JSON

5. HTTP požiadavky/odpovede do textového súboru

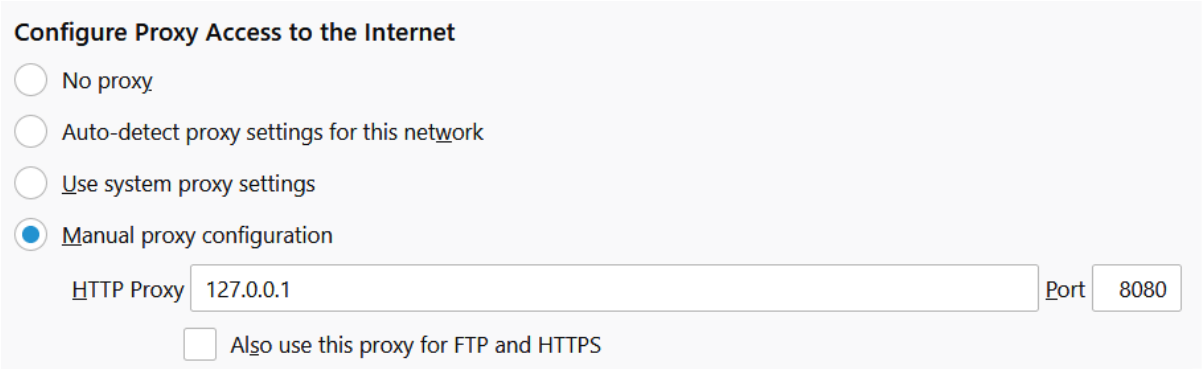
6. URL do textového súboru

Po zvolení výstupu ostáva len zvoliť úložisko.

3.7 Konfigurácia OWASP ZAP

Na úvod je nutné konfigurovať proxy na používanej stanici na localhost(127.0.0.1). Dôvodom tohto kroku je presmerovanie toku HTTP premávky (z portu 80) skrz aplikáciu ZAP kde sa filtruje ďalej definovaným portom. Nastavenie proxy môže prebiehať na ľubovoľnom nepoužívanom porte avšak port 8080 je zaužívaným štandardom. Jedinou podmienkov je totožnosť portu nastavenia proxy v aplikácií ZAP a používaným prehliadačom pre správne vytvorenie pripojenia. Vďaka tomuto spojeniu ZAP dokáže odchytiť HTTP komunikáciu medzi prehliadačom a webovým serverom čo nám ju umožní sledovať a prípadne manipulovať.

3.7.0.1 Firefox nastavenie proxy nájdeme v nastaveniach (pravý horní roh prehliadača), v menu zaklikneme `options`. Do vyhľadávania napíšeme proxy a otvoríme `Network Settings`. Prednastavené je použitie systémového proxy na našej lokálnej stanici to si ukážeme v ďalšom kroku teraz zaklikneme Manuálnu konfiguráciu proxy. Vyplníme pole HTTP proxy pretože budeme filtrovať len premávku z portu 80 teda protokol HTTP. Do poľa zadáme adresu localhostu 127.0.0.1 alebo reťazec localhost. Ako port použijeme číslo 8080 podľa používaného štandardu. Potvrdíme `OK`.



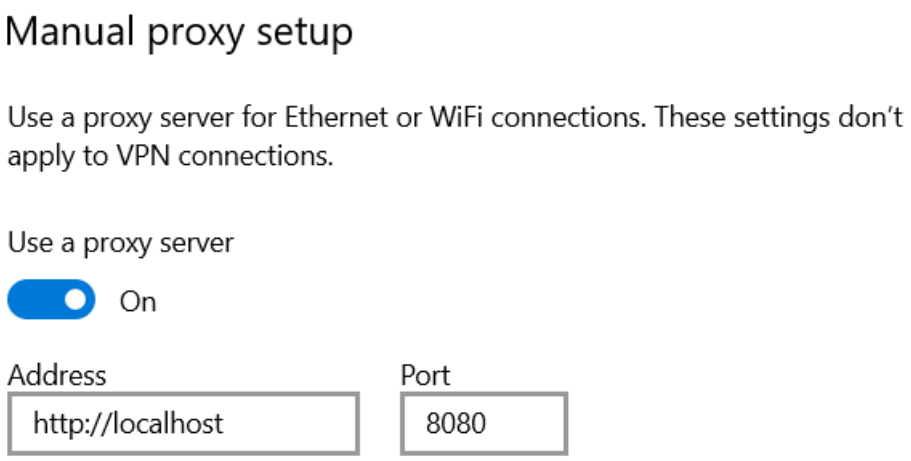
Obr. 8: Nastavenie proxy - Firefox

3.7.0.2 Chrome Vo Windows 10 prehliadač Chrome používa systémovú proxy na lokálne stanici. Prístup k nej môžme získať napríklad:

1. v prehliadači Chrome otvoríme menu (pravý horný roh) a vyberieme *„options“*. V ľavom menu vyberieme *„Advanced“* a po rozšírení menu *„System“*. Na lište vyberieme *„otvoriť nastavenie proxy serveru“*.

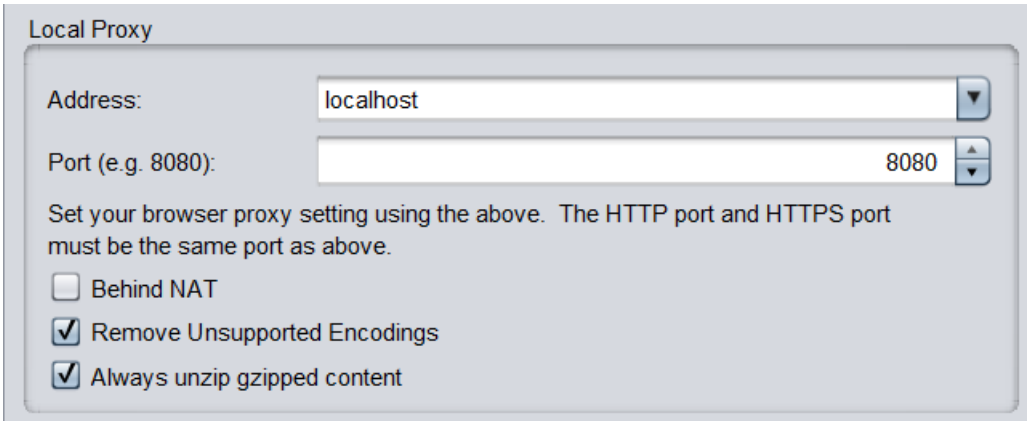
2. Vo Windows 10 klikneme na logo Windows a vyberieme *„nastavenia“*. Z ponuky vyberieme *„Network and Internet“* a z menu prejdeme do *„Proxy“*.

V kategórií Manuálne nastavenie proxy zapneme použitie proxy serveru a nastaví sa adresa na localhost s použitím portu 8080.



Obr. 9: Nastavenie proxy - Chrome, Windows 10

3.7.0.3 OWASP ZAP konfigurácia proxy v aplikácií je posledným krokom. Táto proxy by mala byť nakonfigurovaná automaticky avšak nemusí tomu tak byť. K jej správnej konfigurácií spustíme aplikáciu OWASP ZAP. V hornej lište vyberieme `Tools` a rozlikneme `Options`. V otvorenom okne nájdeme `Local Proxies`. Adresa sa nastaví na localhost a číslo portu musí byť zhodné ako v nastaveniach proxy prehliadača/systému. Ostatné nastavenie môžme ponechať v predvolenom stave pokiaľ sa internetové pripojenie nenachádza za bránou NAT!



Obr. 10: Nastavenie proxy - OWASP ZAP

3.7.1 Certifikát bezpečnosti

Po správnej konfigurácií proxy pripojenia sa komunikácia stala nedôveryhodnou. Web server varuje pred zistením proxy medzi prehliadačom a ním, a pred nezabezpečenou komunikáciou, na základe čoho vyhodnotil komunikáciu ako rizikovú. K zabezpečeniu dôveryhodnosti komunikácie musí byť do systému zakomponovaný SSL bezpečnostný certifikát pre OWASP ZAP. SSL certifikát musí byť vygenerovaný a implementovaný do bezpečnostnej politiky systému.

3.7.1.1 Generovanie SSL certifikátu je možné použitím existujúceho modulu v aplikácií OWASP ZAP na tento účel. V hornej lište prejdeme na `Tools` -> `Options` a vyhľadáme `Dynamic SSL Certificates`. V otvorenom okne zaklikneme `Generate` pre generáciu certifikátu a použijeme `Save` pre uloženie certifikátu na vybrané miesto systéme.

3.7.1.2 Implementácia certifikátu v systéme Windows 10 začína použitím Windows vyhladávania do ktorého sa zadá `certmgr.msc`. Po otvorení okna certifikátov, rozklikneme v ľavom panely `Trusted Root Certification Autorities`. V podadresári klikneme RMB na `Certificates` -> `All Tasks` -> `Import...`. V zobrazenom inštalačnom okne nájdeme vygenerovaný certifikát, zaradíme ho do skupiny `Trusted Root Certification Authorities` a dokončíme inštaláciu. Po tomto kroku je komunikácia s použitím OWASP ZAP ako proxy, zabezpečená.

Po správnej konfigurácií proxy nastavení a implementácií SSL certifikátu stanice je aplikácia pripravená k práci.

4 Záver

Témou mojej bakalárskej práce bolo absolvovanie individuálnej odbornej praxe vo firme XEVOS Solutions, s.r.o . Cieľom tejto práce bolo opísať moje pracovné zaradenie ako aj naplň práce vo firme.

Ďalej som túto prácu venoval práci a problematike penetračného testovania. Učinil som tak preto, pretože táto problematika mi bola osobne najbližšia a aktuálna. Pred začatím praktickej časti práce bolo nutné objasniť základnú teóriu tohto odvetvia ako aj niektoré pojmy, metodiky a použité postupy. Po tomto kroku som prešiel na časť praktickú, kde som sa venoval testovaniu webového prostredia podľa metodiky OWASP, s použitím open source nástrojov. Výstupom praktickej časti sú postupy jednotlivých krokov a reporty práce.

Záverom práce je moje zhodnotenie použitých teoretických a praktických znalostí poskytnuté štúdiom, znalosti ktoré som si musel doplniť, a nakoniec moje dosiahnuté výsledky a celkové zhodnotenie praxe.

4.1 Teoretické a praktické znalosti a schopnosti získané v priebehu štúdia uplatnené študentom v priebehu odbornej praxe

Pri výkone odbornej individuálnej praxe som, priamo aj nepriamo, nadväzoval na niekoľko predmetov absolvovaných počas môjho bakalárskeho štúdia. Pri práci na penetračnom testovaní to boli konkrétne Telekomunikační sítě, Počítačové sítě, Přenos dat. Tieto predmety mi poskytli znalosti o rôznych internetových a komunikačných protokolov. Pri práci s interným systémom a databázou mi bol nápomocný predmet Úvod do databázových systémů, a Úvod do teoretické informatiky pri pochopení a tvorení regulárnych výrazov. Pri účasti na vývoji webovej aplikácie som využil znalosti z niekoľkých predmetov ako Algoritmy I/II a Programování I/II.

4.2 Znalosti a schopnosti chýbajúce študentovi v priebehu odbornej praxe

Pri svojom pôsobení vo firme som narážal na nedostatky a medzery vo svojich schopnostiach. Tieto nedostatky boli väčšinovo charakteru praktického. Pri práci na vývoji aplikácie to boli konkrétne nové programy a technológie, avšak verím, že to bol rozdiel spôsobení odlišnosťou práce k môjmu odboru Telekomunikační technika. Pri penetračnom testovaní, práca v nových prostrediach a odlišné vnímanie aplikačného prostredia. A nakoniec pri vytváraní scriptových príkaz pre správu zariadení v programe Hyper-V, to bola nutnosť vzdelať sa v scriptovaní príkazov.

4.3 Dosiahnuté výsledky v priebehu odbornej praxe a jej celkové zhodnotenie

Absolvovanie odbornej praxe vo firme XEVOS hodnotím veľmi kladne. Pomohla mi porovnať kvalitu výuky a požiadavky profesionálnej sféry. Zoznámil som sa s chodom interného prostredia IT firmy ako aj postupmi pri riešení nových zákaziek a problémov. Bola mi poskytnutá mož-

nosť pracovať s novými technológiami a kvalitným personálom. Táto prax určite posunula moje osobné schopnosti a prehĺbila záujem o toto odvetvie, či už štúdijný alebo pracovný.

Literatúra

[1] Penetration Testing | IT Governance UK. Itgovernance.co.uk [online]. Cambridgeshire: IT Governance, c2003-2020 [cit. 2020-04-28]. Dostupné z: https://www.itgovernance.co.uk/penetration-testing

[2] DIETERLE, Daniel W. Intermediate Security Testing with Kali Linux 2. Lexington, KY: Cyberarms, 2015. ISBN 978-1516945863

[3] SELECKÝ, Matúš. Penetrační testy a exploitace. Brno: Computer Press, 2012. ISBN 978-80-251-3752-9.

[4] SAINDANE, Manish. Penetration testing – A Systematic Approach. Http://www.infosecwriters.com [online]. 8.11.2015 [cit. 2020-04-23]. Dostupné z: http://www.infosecwriters.com/articles/2015/08/11/penetration-testing-%E2%80%93-systematic-approach

[5] Threat Modelling. Softscheck [online]. Sankt Augustin: softScheck, 2016 [cit. 2020-04-28]. Dostupné z: https://www.softscheck.sg/threat-modelling/

[6] HTTP Header. Techopedia [online]. Edmonton, Canada: Janalta Interactive, 2001, 1.5.2013 [cit. 2020-04-28]. Dostupné z: https://www.techopedia.com/definition/27178/http-header

[7] Nmap Network Scanning: Nmap Reference Guide. Nmap [online]. Kalifornia: Gordon Lyon, 1997 [cit. 2020-04-28]. Dostupné z: https://nmap.org/

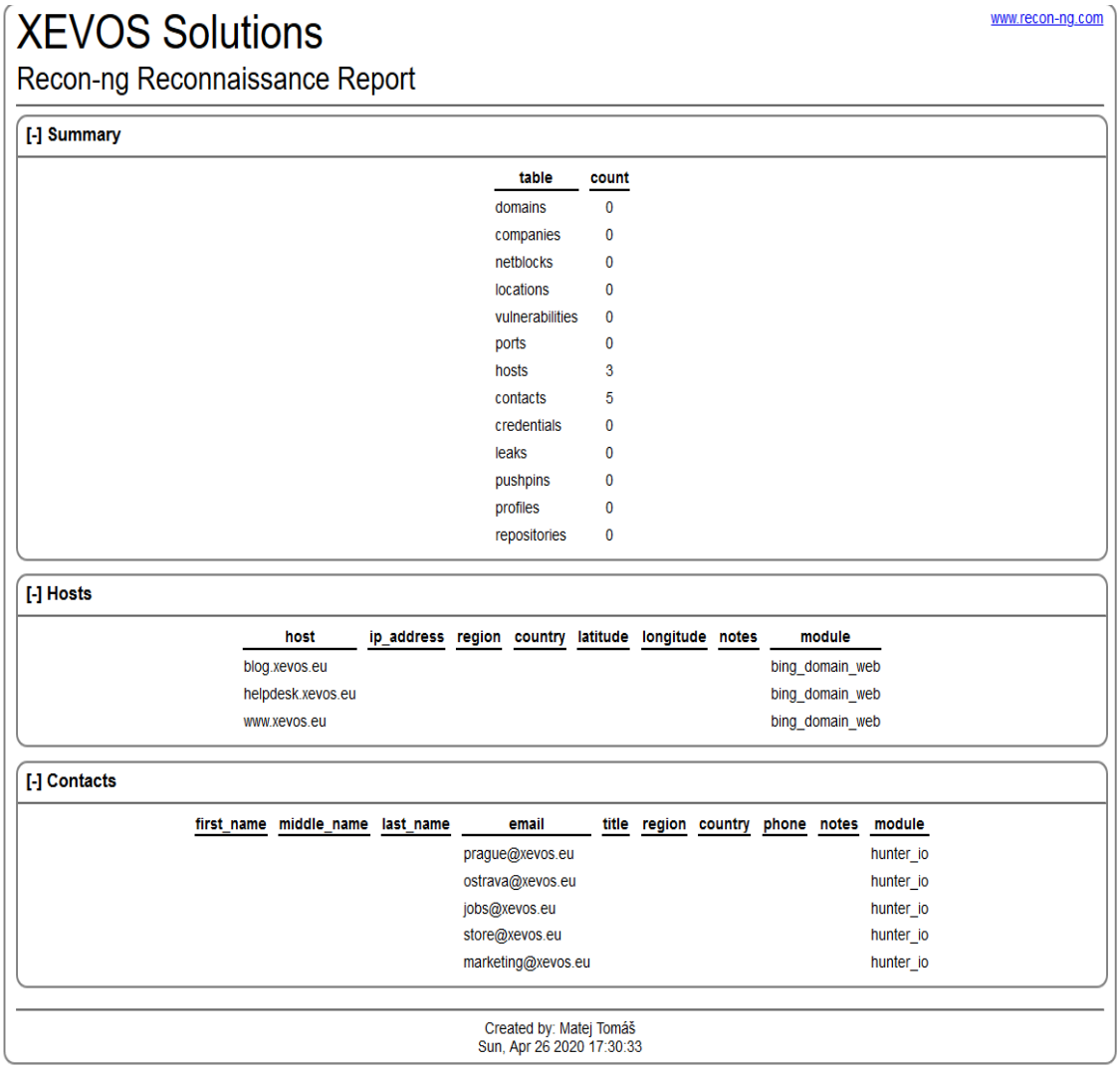
[8] Theharvester Package Description. Tools.kali [online]. New York: OffSec Services Limited, c2020 [cit. 2020-04-28]. Dostupné z: https://tools.kali.org/information-gathering/theharvester

[9] Recon-ng Package Description. Tools.kali [online]. New York: OffSec Services Limited, c2020 [cit. 2020-04-28]. Dostupné z: https://tools.kali.org/information-gathering/recon-ng

[10] DMitry: DMitry Package Description. Tools.kali [online]. New York: OffSec Services Limited, c2020 [cit. 2020-04-28]. Dostupné z: https://tools.kali.org/information-gathering/dmitry

[11] XSS Filter Evasion Cheat Sheet. Owasp.org [online]. Maryland: Curphey, Groves, 2001 [cit. 2020-05-11]. Dostupné z: https://owasp.org/www-community/xss-filter-evasion-cheatsheet

A Recon-ng príklad reportu.html



XEVOS Solutions

www.recon-ng.com

Recon-ng Reconnaissance Report

[-] Summary

table	count
domains	0
companies	0
netblocks	0
locations	0
vulnerabilities	0
ports	0
hosts	3
contacts	5
credentials	0
leaks	0
pushpins	0
profiles	0
repositories	0

[-] Hosts

host	ip_address	region	country	latitude	longitude	notes	module
blog.xevos.eu							bing_domain_web
helpdesk.xevos.eu							bing_domain_web
www.xevos.eu							bing_domain_web

[-] Contacts

first_name	middle_name	last_name	email	title	region	country	phone	notes	module
			prague@xevos.eu						hunter_io
			ostrava@xevos.eu						hunter_io
			jobs@xevos.eu						hunter_io
			store@xevos.eu						hunter_io
			marketing@xevos.eu						hunter_io

Created by: Matej Tomáš
Sun, Apr 26 2020 17:30:33

Obr. 11: Recon-ng generovaný report.html

B Dmitry .txt výstup

Príklad výstupu nástroja DMITRY. Adresy, názvy a identifikačné reťazce boli pozmenené , a redigované z dôvodu obsahu citlivých informácií.

```
HostIP: 188.210.180.205
HostName: test.domena.net

Gathered Inet-whois information for 188.210.180.205
---------------------------------


inetnum:        188.210.180.20 - 188.210.180.235
netname:        MATEJNET-CUSTOMERS
descr:          MatejNet
country:        CZ
admin-c:        LD5390-RIPE
tech-c:         LD5390-RIPE
status:         ASSIGNED PA
mnt-by:         cz-matejnet-1-mnt
created:        2017-10-10T07:00:36Z
last-modified:  2018-03-17T20:56:10Z
source:         *redigované*
org:            ORG-JS1310

organisation:   ORG-JS1310
org-name:       MATEJNet
org-type:       LIR
address:        *redigované*
address:        *redigované*
admin-c:        LD5390-RIPE
tech-c:         LD5390-RIPE
mnt-ref:        cz-matejnet-1-mnt
mnt-by:         cz-matejnet-1-mnt
created:        2017-07-04T06:27:32Z
last-modified:  2017-07-04T09:52:54Z
source:         RIPE # Filtered
phone:          +420*********

person:         *redigované*
```

```
address:        *redigované*
address:        *redigované*
phone:          +420*********
mnt-by:         cz-matejnet-1-mnt
created:        2017-07-04T06:27:32Z
last-modified:  2017-07-04T06:27:32Z
source:         RIPE

% Information related to '188.210.180.0/22AS49573'

route:          188.210.180.0/22
origin:         *redigované*
mnt-by:         cz-matejnet-1-mnt
created:        2017-07-11T16:41:03Z
last-modified:  2018-01-10T13:28:42Z
source:         RIPE
descr:          MatejNet

% This query was served by the RIPE Database Query Service
version 1.97 (HEREFORD)
```

C Nmap http-headers výpis

```
80/tcp  open  http

| http-headers:

|   Cache-Control: no-cache, no-store

|   Pragma: no-cache

|   Transfer-Encoding: chunked

|   Content-Type: text/html; charset=utf-8

|   Expires: Thu, 01 Jan 1970 00:00:00 GMT

|   Server: Microsoft-IIS/10.0

|   Set-Cookie: Identity.External=; expires=Thu, 01 Jan 1970 00:00:00 GMT;
    path=/; samesite=lax; httponly

|   Set-Cookie: .AspNetCore.Antiforgery.oICvOtI4CJk=CfDJ8JPU4T
    OkOq5Mjc8Z79iKT5l3omVLkNPO42oKTo2UeIww1Z_SK075thivg3bIaFTugJg8De
    UImbgzeTy8qXneC9TaG_eJrbtzFbok-
        EOQPEZufA2YyvcJn41iUKutFrh44F3HAG7srEwdfk7UbQH4NZY; path=/; samesite=
        strict; httponly

|   Set-Cookie: .AspNetCore.Mvc.CookieTempDataProvider=; expires=Thu, 01 Jan
    1970 00:00:00 GMT; path=/; samesite=lax; httponly

|   X-Frame-Options: SAMEORIGIN

|   X-Powered-By: ASP.NET

|   Date: Tue, 28 Apr 2020 13:30:20 GMT

|   Connection: close

|
```

```
|_  (Request type: HEAD)
```

```
8008/tcp open  http

| http-headers:

|   Location: https://test.domena.net:8010/

|   Connection: close

|   X-Frame-Options: SAMEORIGIN

|   X-XSS-Protection: 1; mode=block

|   X-Content-Type-Options: nosniff

|   Content-Security-Policy: frame-ancestors

|

|_  (Request type: GET)
```

D Nmap výpis základných príkazov

Popis	Príkaz
1 IP adresa	`nmap 192.168.1.1`
Scan hosta	`nmap www.example.com`
Rozsah adries	`nmap 192.168.1.1-48`
Subnet	`nmap 192.168.1.0/20`
Adresy zo súboru	`nmap -iL list-of-ips.txt`
Port	`nmap -p 80 192.168.1.1`
Rozsah portov	`nmap -p 1-100 192.168.1.1`
Fast-100 bežných portov	`nmap -F 192.168.1.1`
Všetky porty	`nmap -p- 192.168.1.1`
TCP connect	`nmap -sT 192.168.1.1`
TCP SYN scan	`nmap -sS 192.168.1.1`
UDP port scan	`nmap -sU -p 123,161,162 192.168.1.1`
Služby a OS	`nmap -A 192.168.1.1`
Štandardné služby	`nmap -sV 192.168.1.1`
HTTP page title	`nmap --script=http-title 192.168.1.0`
HTTP záhlavie služieb	`nmap --script=http-headers 192.168.1.0`
Webové aplikácie podľa ciest	`nmap --script=http-enum 192.168.1.0`
Vyhľadávanie whois	`nmap --script whois-ip 192.168.1.0`
Test zraniteľností serveru	`nmap -Pn --script vuln 192.168.1.0`
Malware/backdoor scan	`nmap -sV --script=http-malware-host 192.168.1.0`
Google malware scan	`nmap -p80 --script http-google-malware example.com`

Tabuľka 2: Nmap základné príkazy

E OWASP ZAP report automatizovanej analýzy

Výpis prvých 3 strán automatizovanej analýzy zraniteľností.

Celý zdroj v prílohe zapReport.pdf

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	11
Low	13
Informational	17

Alert Detail

Medium (Medium)	Cross-Domain Misconfiguration
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server
URL	https://gyrocode.github.io/jquery-datatables-checkboxes/1.2.11/js/dataTables.checkboxes.min.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://gyrocode.github.io/jquery-datatables-checkboxes/1.2.11/css/dataTables.checkboxes.css
Method	GET
Evidence	Access-Control-Allow-Origin: *
Instances	2
Solution	Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.
Other information	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
Reference	http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html
CWE Id	264
WASC Id	14
Source ID	3

Medium (Medium)	Cross-Domain Misconfiguration
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server
URL	https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-1.10.20/af-2.3.4/b-1.6.1/b-colvis-1.6.1/b-html5-1.6.1/b-print-1.6.1/fc-3.3.0/fh-3.1.6/r-2.2.3/datatables.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://cdn.datatables.net/v/bs4/jszip-2.5.0/dt-1.10.20/af-2.3.4/b-1.6.1/b-colvis-1.6.1/b-html5-1.6.1/b-print-1.6.1/fc-3.3.0/fh-3.1.6/r-2.2.3/datatables.css
Method	GET
Evidence	Access-Control-Allow-Origin: *
Instances	2
Solution	Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.
Other information	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
Reference	http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html
CWE Id	264
WASC Id	14

Source ID	3

Medium (Medium)	Application Error Disclosure
Description	This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.
URL	https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.1.36/pdfmake.js
Method	GET
Evidence	internal error
Instances	1
Solution	Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.
Reference	
CWE Id	200
WASC Id	13
Source ID	3

Medium (Medium)	Cross-Domain Misconfiguration
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server
URL	https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.3/Chart.bundle.min.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.3/Chart.min.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.1.36/vfs_fonts.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.1.36/pdfmake.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
Instances	4
Solution	Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.
Other information	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
Reference	http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html
CWE Id	264
WASC Id	14
Source ID	3

Medium (Medium)	Cross-Domain Misconfiguration
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server
URL	https://cdn.jsdelivr.net/npm/promise-polyfill@7.1.0/dist/promise.min.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://cdn.jsdelivr.net/npm/html-to-pdfmake/docs/browser.js
Method	GET
Evidence	Access-Control-Allow-Origin: *
Instances	2

Solution	Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.
Other information	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
Reference	http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html
CWE Id	264
WASC Id	14
Source ID	3

Medium (Medium)	**Cross-Domain Misconfiguration**
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server
URL	https://fonts.gstatic.com/s/poppins/v9/pxiByp8kv8JHgFVrLEj6Z1JlFc-K.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiByp8kv8JHgFVrLGT9Z1xlFQ.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiByp8kv8JHgFVrLGT9Z1JlFc-K.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiEyp8kv8JHgFVrJJfecg.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiByp8kv8JHgFVrLCz7Z1xlFQ.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiEyp8kv8JHgFVrJJnecmNE.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiByp8kv8JHgFVrLEj6Z1xlFQ.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
URL	https://fonts.gstatic.com/s/poppins/v9/pxiByp8kv8JHgFVrLCz7Z1JlFc-K.woff2
Method	GET
Evidence	Access-Control-Allow-Origin: *
Instances	8
Solution	Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.
Other information	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
Reference	http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html
CWE Id	264
WASC Id	14
Source ID	3

Medium (Medium)	**Cross-Domain Misconfiguration**
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server